Department of Computer Engineering

CENG350 Software Engineering

Software Requirements Specification for FarmBot

Group 6

By

Berk Ulutaş

Koray Özgür

Saturday 6th April, 2024

Contents

List of Figures

List of Tables

Revision History

Revision	**Date**	**Authors**	**Description**
0.1	15.03.2024	Berk Ulutaş, Koray Özgür	Initalization
1.0	29.03.2024	Berk Ulutaş, Koray Özgür	Part-1
2.0	06.04.2024	Berk Ulutaş, Koray Özgür	Final

Table 1: Revision History

1. Introduction

1.1 Purpose of the System

FarmBot is an open-source, fully customizable robotic farming system. With its remote controllability and ability to automate food-growing tasks, FarmBot revolutionizes agriculture by streamlining processes and making farming more accessible and efficient. By enabling users to automate repetitive tasks, FarmBot empowers individuals and communities to cultivate their own crops with precision and ease.

1.2 Scope

- FarmBot control system, which includes a web-based interface for remotely controlling and managing the robotic farming operations.
- The FarmBot control system will allow users to plan, schedule, and monitor various farming tasks such as planting seeds, watering plants, monitoring plant health, and controlling other aspects of the growing process. It will provide a user-friendly interface accessible via web browsers on different devices.

- The application of the FarmBot control system involves automating and optimizing farming processes, aiming to make agriculture more accessible, efficient, and sustainable. By enabling remote control and automation of tasks, FarmBot reduces labor requirements, minimizes resource wastage, and maximizes yields. The system's benefits include increased productivity, reduced manual labor, and improved precision in farming operations.
- The FarmBot control system will align with user needs and preferences, providing customizable features to accommodate various farming setups and requirements. Additionally, it will integrate with sensors and other hardware components as necessary to monitor and control the growing environment effectively.

1.3 System Overview

1.3.1 System Perspective

The FarmBot system operates at the intersection of user needs, environmental data, and agricultural knowledge sourced from open platforms like openfarm.cc. From a system perspective, FarmBot serves as a centralized information hub, receiving commands and farm designs from users and delivering appropriate feedback in the form of sensor data, photos, and logs. This bidirectional communication enables users to monitor and manage their farm operations remotely, empowering them with real-time insights and control over the farming process.

FarmBot operates within various constraints and interfaces to facilitate efficient and effective automated farming processes. FarmBot interacts with users, the farm environment, and openfarm.cc, facilitating communication between them. Intuitive web or mobile interfaces allow users to control FarmBot and access farming data. FarmBot communicates with sensors, actuators, and robotic arms for data gathering and task execution. Utilizing protocols like Wi-Fi or Ethernet, FarmBot connects with external entities and services. It adapts to different farm environments, crops, and user preferences. These aspects will be further elaborated on in the subsequent sections.

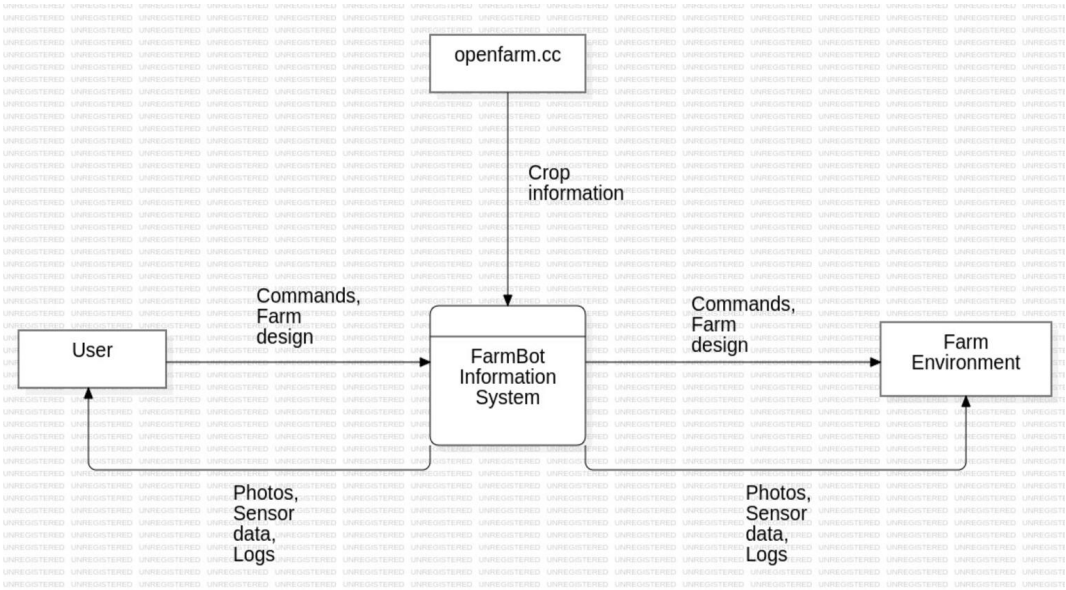

Figure 1.1: Context Diagram

1.3.1.1 System Interfaces

Camera Management System Interface: To manage and control the cameras used in the FarmBot system for monitoring the farm environment. This interface enables FarmBot to access and control camera devices deployed across the farm. It includes functionalities such as adjusting camera angles, capturing images, and streaming live footage for monitoring plant growth and environmental conditions.

Sensors Management System Interface: To manage sensors deployed within the FarmBot system to collect data about the farm environment. This interface allows FarmBot to interact with various types of sensors, such as soil moisture sensors, temperature sensors, and humidity sensors. It includes functionalities for configuring sensor settings, reading sensor data, and triggering actions based on sensor readings. Compliance with MQTT communication protocol.

Seed Inventory Management System Interface: To track and manage the inventory of seeds available for planting within the FarmBot system. This interface provides functionalities for maintaining a database of seed inventory, including details such as seed types, quantities, and expiration dates. It allows users to view available seeds, add new seed to the inventory, and update inventory levels based on planting activities.

openfarm.cc API Interface: To access crop related information and recommendations from the openfarm.cc platform. This interface enables FarmBot to retrieve data such as crop planting guides and recommended planting dates from the openfarm.cc API

Command Management System Interface: To process and execute commands issued by users or automated processes. This interface manages the flow of commands within the FarmBot system. Information exchanged includes task instructions, command parameters, and execution status updates.

1.3.1.2 User Interfaces

Map Interface

FarmBot user can see everything that has been done in the garden through the map interface. User can see plants that are currently set into the garden, the watering nozzle, seeder. Clicking to a object will open its corresponding interface. For example, if user clicks a plant object, eventually corresponding plant's interface will be open.

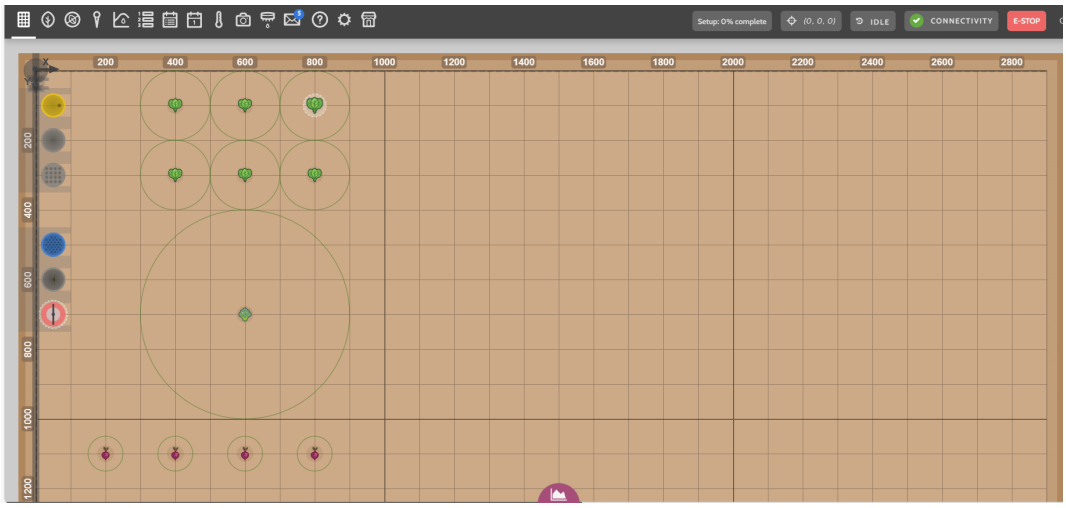

Figure 1.2: Map Interface

Plant Interface

The plant user interface allows you to effortlessly view and manage all the plants in your garden. Simply hovering over a plant in the list will highlight its location on the map and vice versa. Clicking a plant will open up the plant details panel where you can edit it. Pressing the plant + button will allow you to search for and add new crops to your garden. Also selecting spesific plant will open a new tab to edit clicked plant.

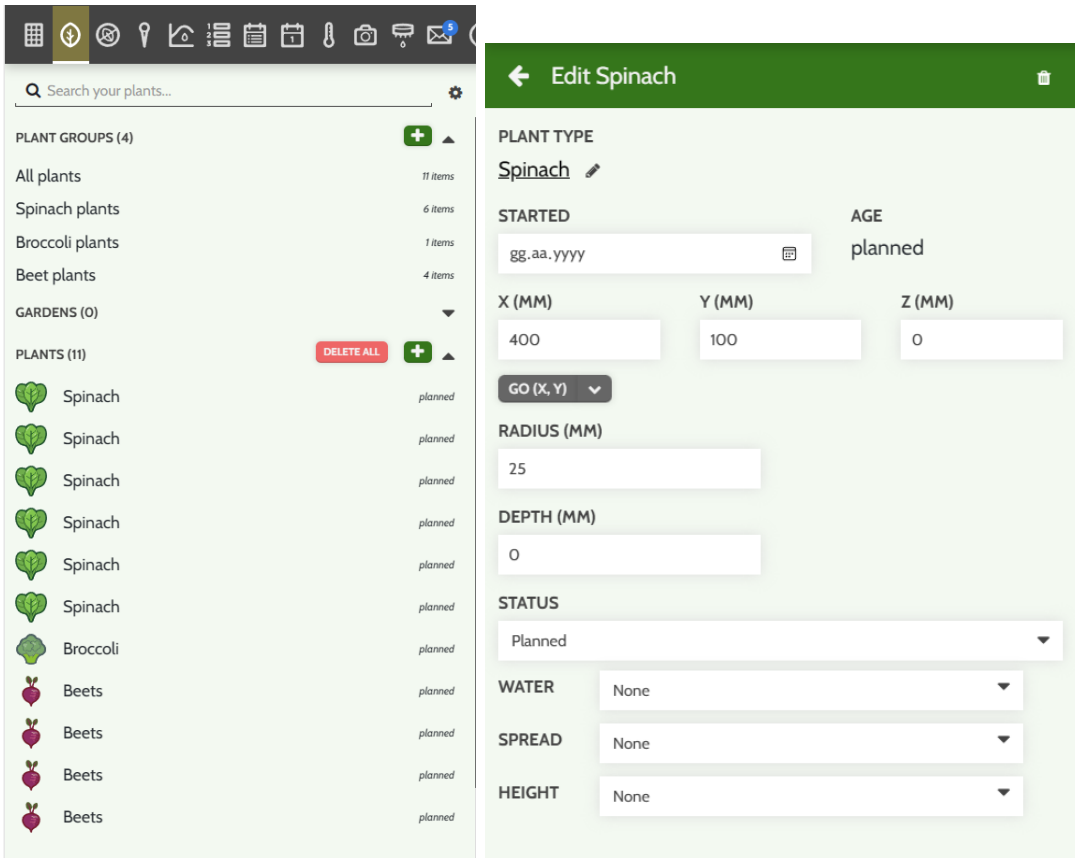

Figure 1.3: Plant Interface

Weed Interface

By using the weed interface of FarmBot, user can view and handle all of the weeds in the garden. Mousing over a weed in the list will highlight it in the map and vice versa. Newly detected weeds will be listed in the PENDING category, requiring periodic approval to be listed in the ACTIVE category. When weeds are addressed by FarmBot, it can be moved to the REMOVED category with a Mark As sequence command. Clicking a weed will open up the weed details panel where you can manually edit it.

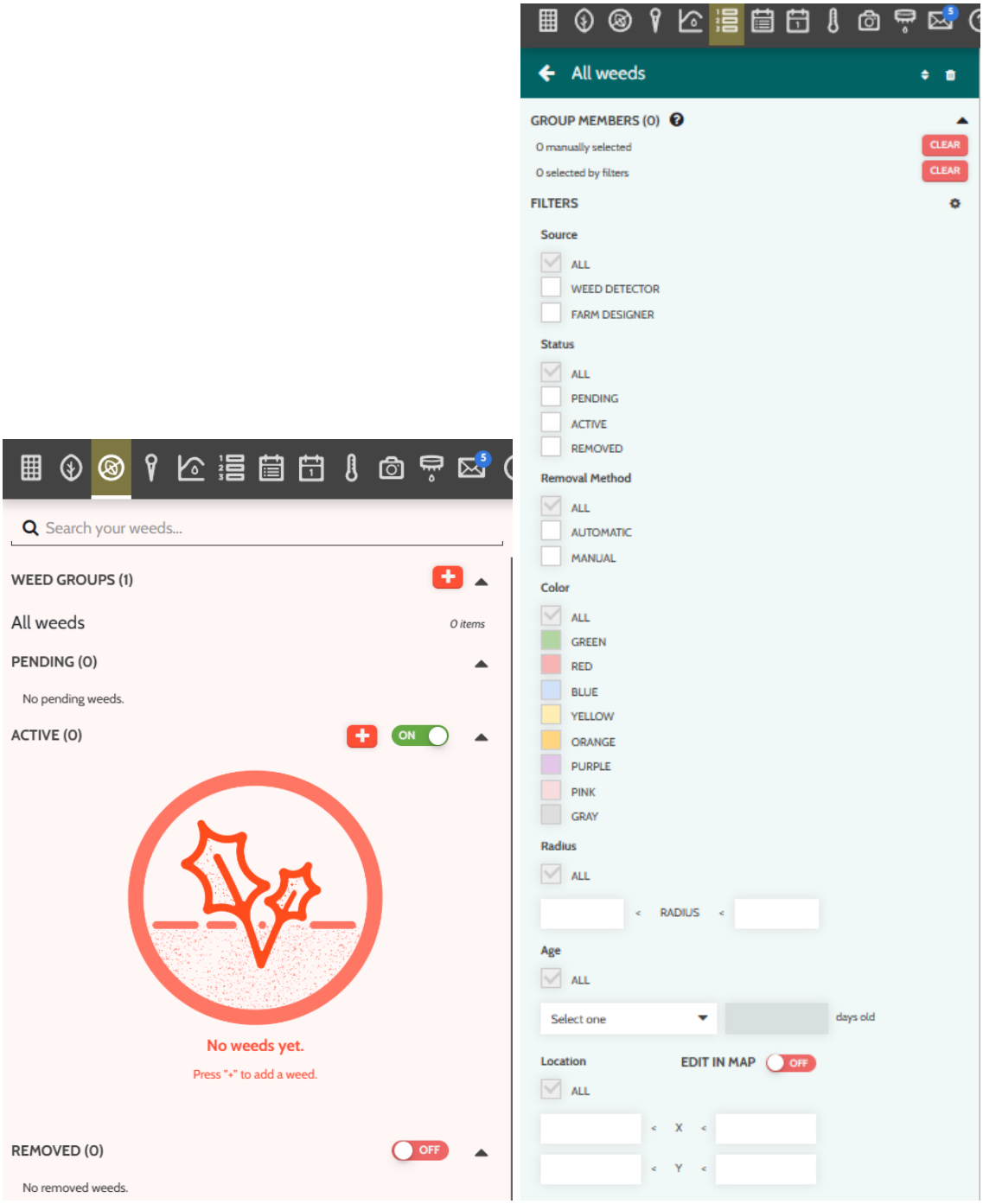

Figure 1.4: Weed Interface

Point Interface

Through the point interface of FarmBot, user can view and manipulate all of the points in the garden, including measured soil height points, via point interface. Mousing over a point in the list will highlight it in the map and conversely, highlighting a point on the map will reflect in the list. Clicking a point will open up the point details panel where you can edit it. Furthermore, pressing the point + button will allow you to add a point or a grid of points to the map. User can set color and radius of points manually based on some restrictions such as age or location in the map if required.

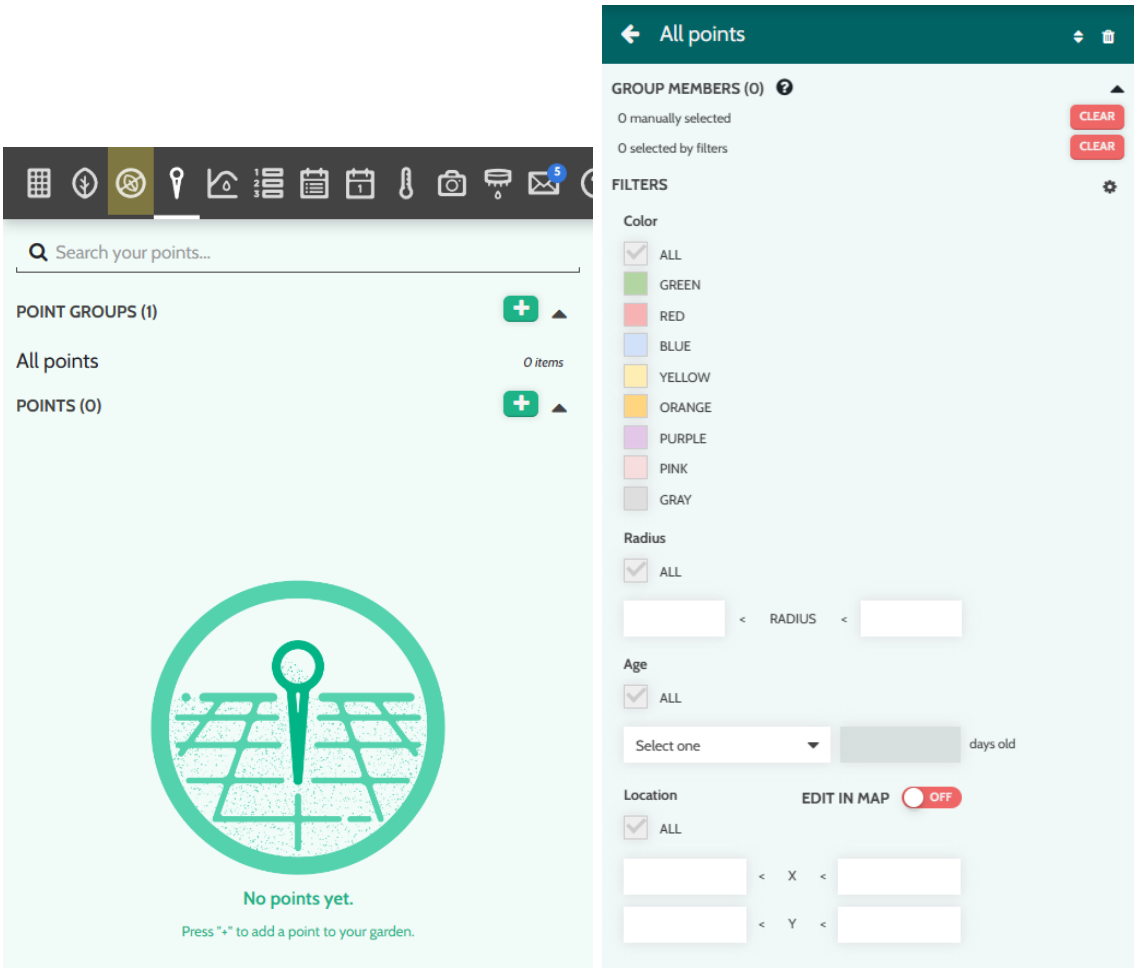

Figure 1.5: Point Interface

Curve Interface

Through curve interface on FarmBot, users can view and regulate water, spread, and height curves for their plants efficiently. By simply pressing the + button, users can generate new curves. This interface allows users to fine-tune curves according to their specific gardening needs. With a user friendly interface, individuals can effortlessly navigate through their curves, ensuring optimal growth and health for their plants.

Figure 1.6: Curve Interface

Sequence Interface

With sequence panel, user can view and manage all of the sequences. Clicking on a sequence opens up the sequence editor panel that enables users to make necessary modifications and adjustments. Additionally, pressing the + button will create a new sequence.

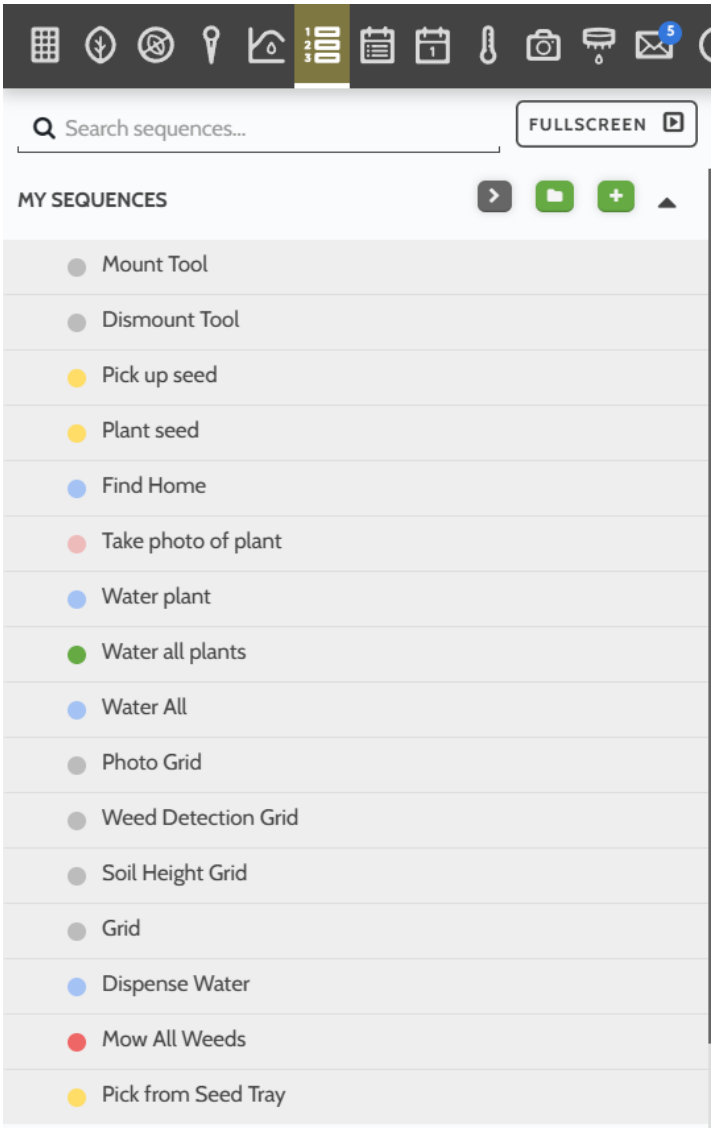

Figure 1.7: Sequence Interface

Regimen Interface

The regimen interface in FarmBot allows users to view and manage all of their regimens which allow them to schedule multiple sequences based on the age of a plant. Clicking on a regimen opens the regimen editor panel where user can make any neccessary changes. Pressing the + button will create a new regimen.

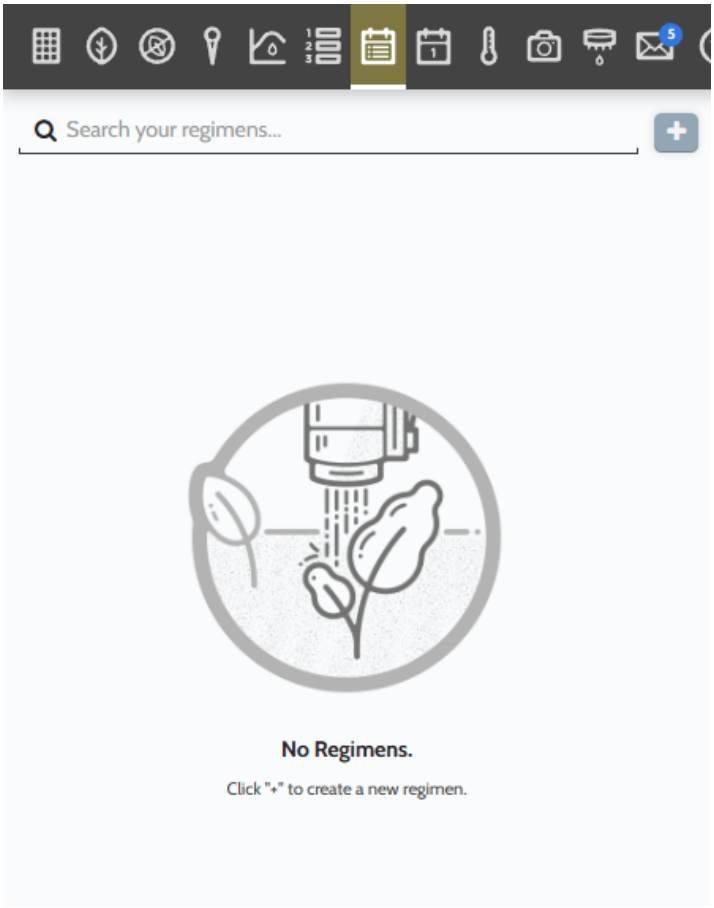

Figure 1.8: Regimen Interface

Event Interface

The event interface of FarmBot allows users to view and manage all of their scheduled sequences and regimens. Clicking an event will open up the event details panel where user can edit it. Pressing the + button will create a new event.

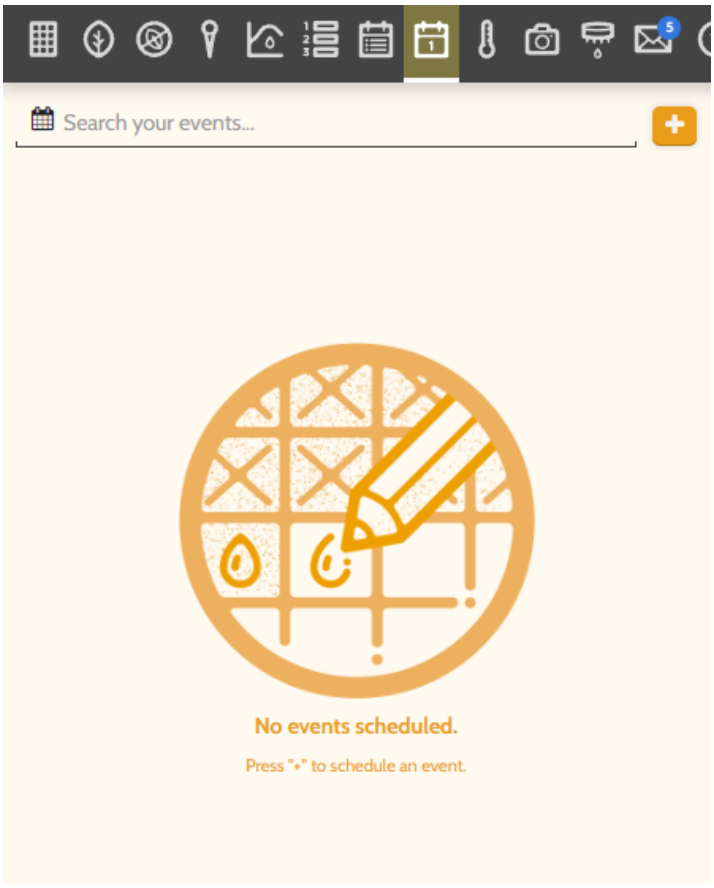

Figure 1.9: Event Interface

Sensor Interface

The sensor interface of FarmBot allows users manage their sensors. Users can access current sensor values by clicking on the "Read Sensor" buttons. Also with this interface, users can create and view historical sensor readings in the Sensor History section.

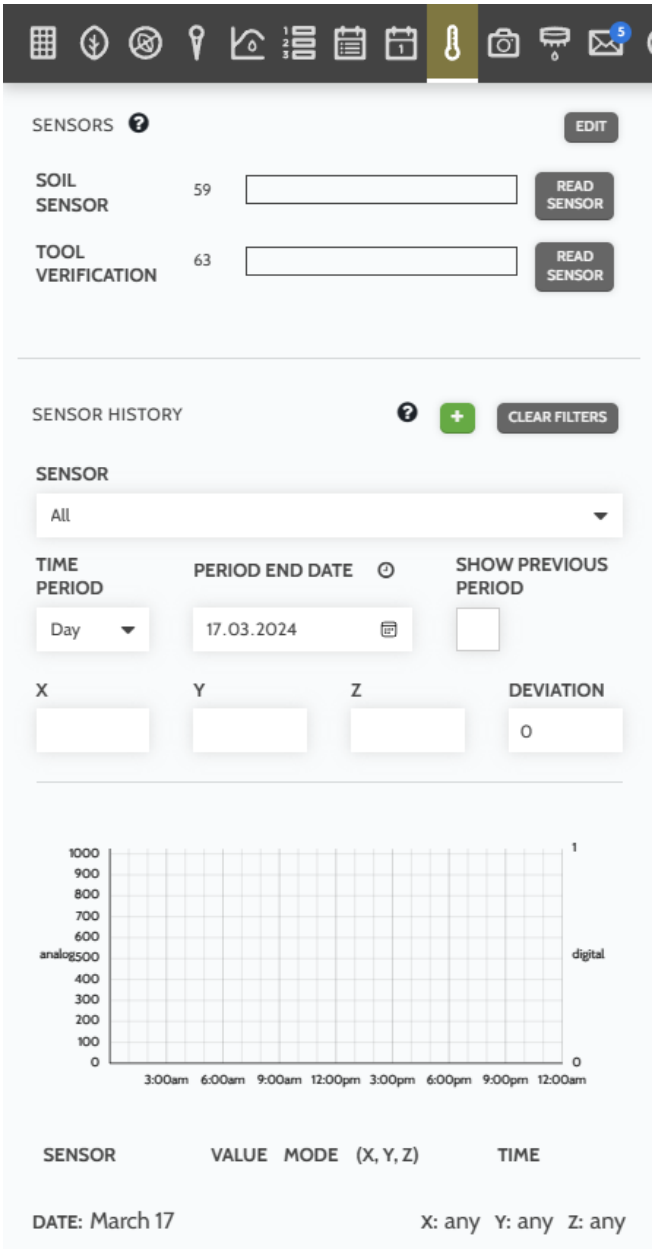

Figure 1.10: Sensor Interface

Photo Interface

With the help of photo interface of FarmBot, users can view and organize all of the photos of their garden. Additionally, users can use this interface to calibrate FarmBot's camera and weed detection settings to optimize the performance for desired lighting conditions and soil type.

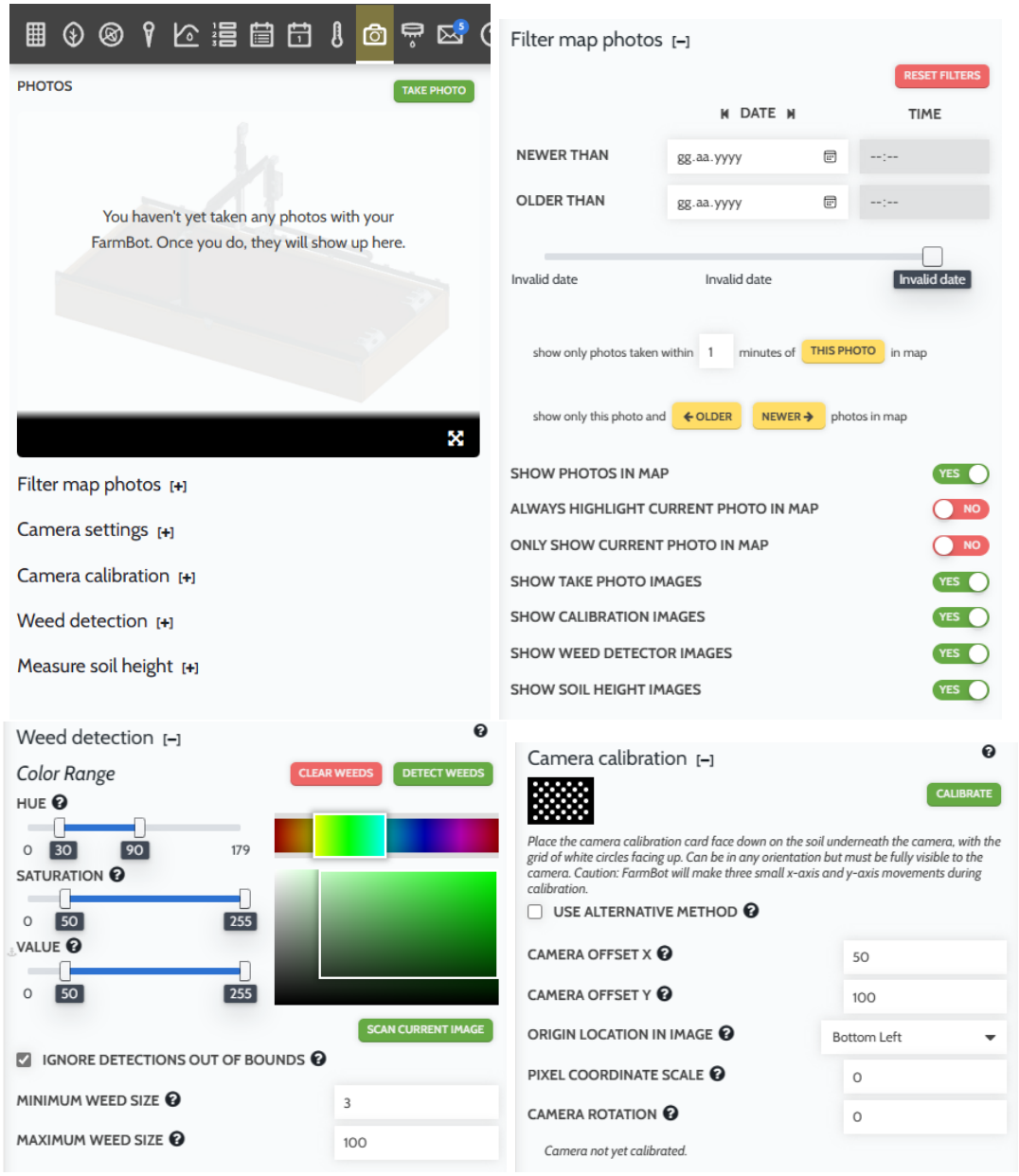

Figure 1.11: Photo Interface

Tool Interface

The tool interface on FarmBot helps users to view and manage all of the tools, seed containers, and slots in their garden. Clicking an item will open up the details panel where user can make necessary edits and adjustments. Pressing the + buttons will allow user to add new items.

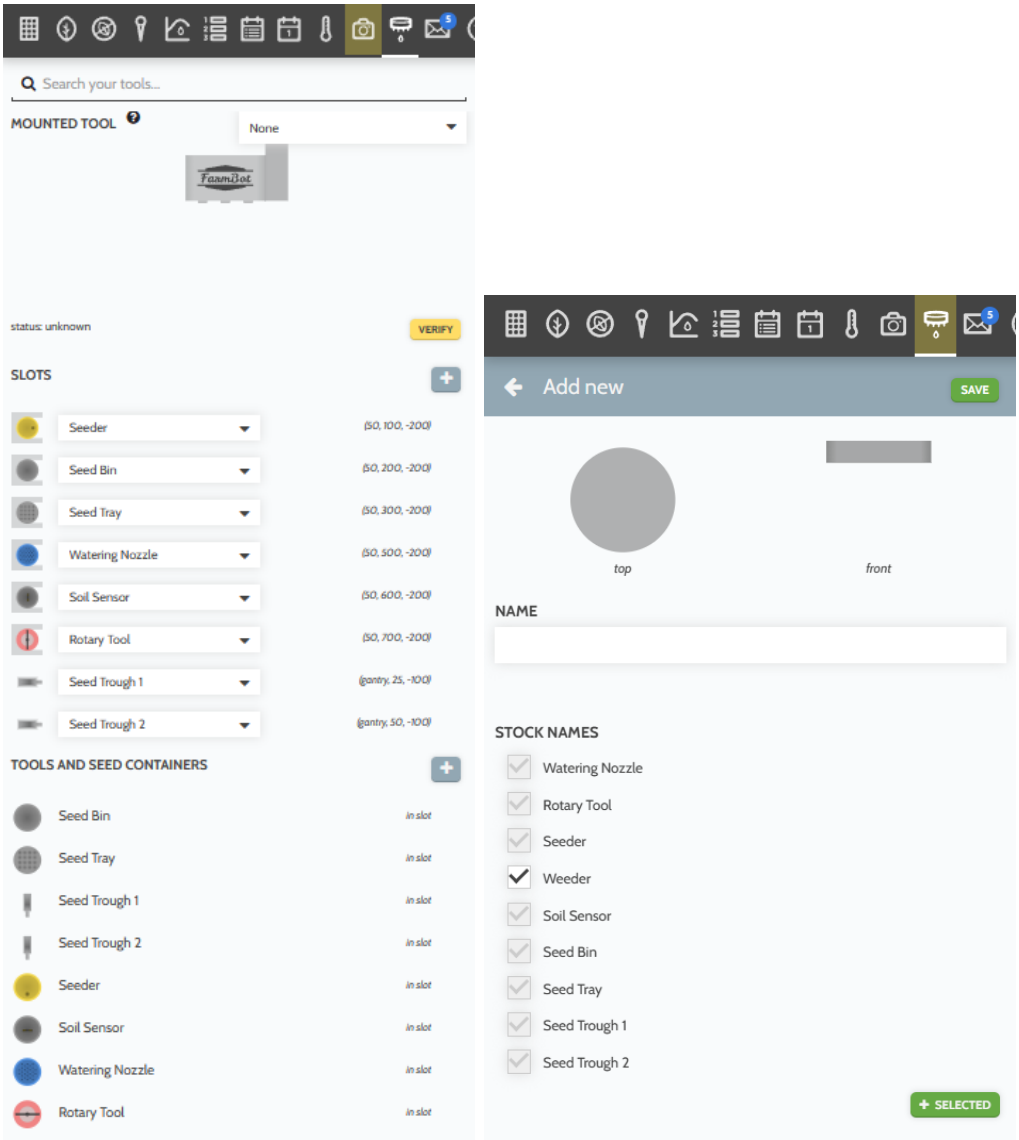

Figure 1.12: Tool Interface

Setting Interface

Users can view and manage all FarmBot settings and account settings according to the desired properties via this setting interface. Users can check the tooltips and documentation to get information about each setting and their functionalities.

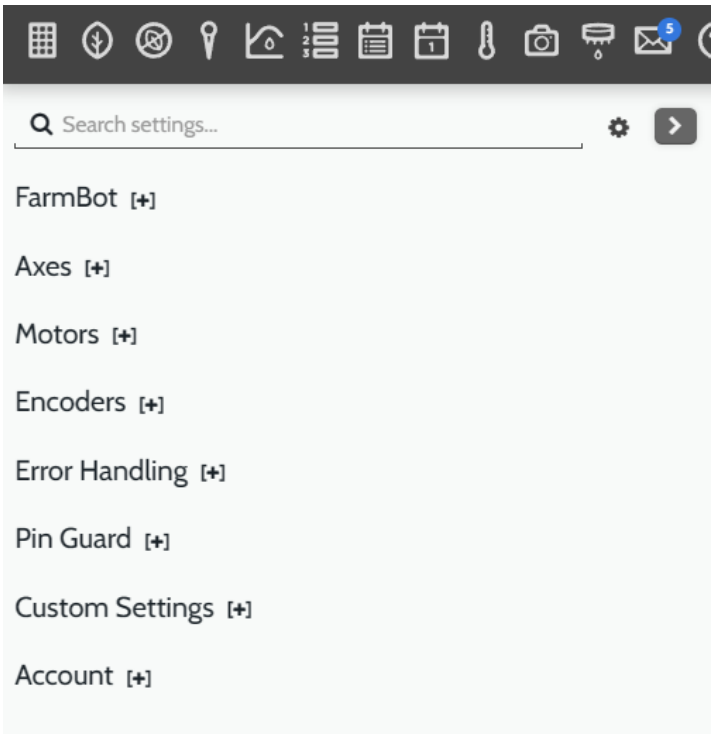

Figure 1.13: Setting Interface

Control Interface

Through the control interface of FarmBot, users can control their FarmBot in real time by pressing the movement arrow buttons, toggling peripherals, and executing sequences to manage operations. They can also view current status information about their FarmBot such as its position and an optional webcam stream.

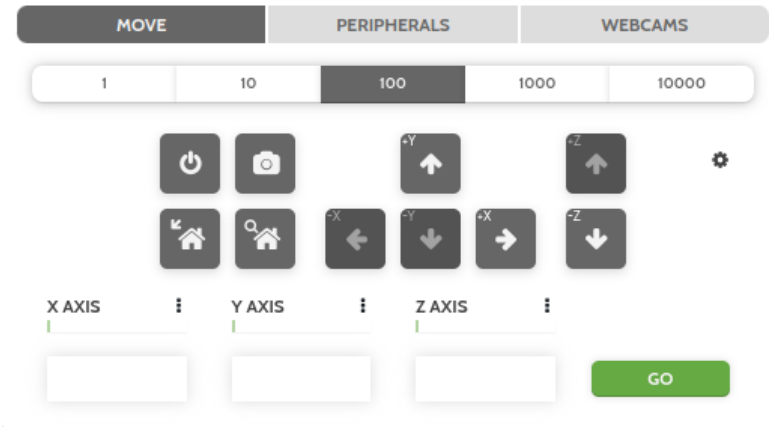

Figure 1.14: Control Interface

Connectivity Interface

Through FarmBot's connectivity interface, users can view information about the connection status between their web browser, FarmBot, and the FarmBot web app servers. Green checkmark indicates that FarmBot is online and ready to execute tasks.

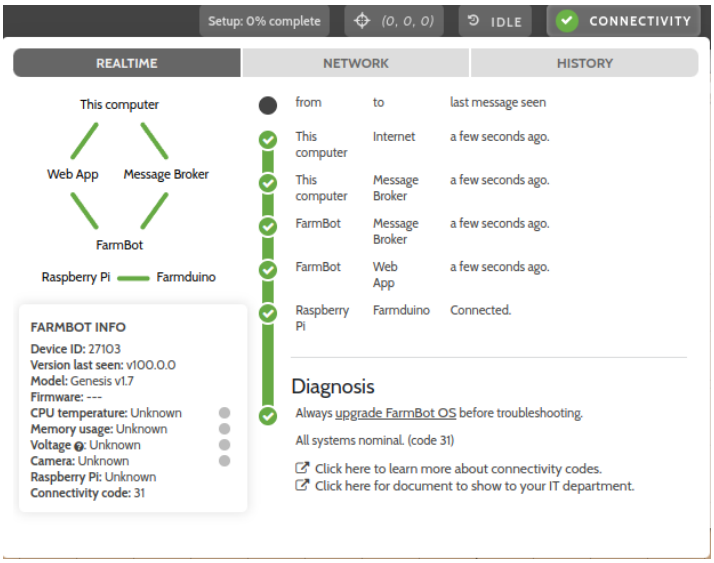

Figure 1.15: Connectivity Interface

1.3.1.3 Hardware Interfaces

Webcam Interface: FarmBot Express supports webcams, allowing users to capture images and video footage of their farm environment. FarmBot supports any plug-and-play USB cameras and the official Raspberry Pi camera. The software is designed to work seamlessly with standard USB webcam protocols and drivers, ensuring easy setup and compatibility.

Raspberry Pi Interface: FarmBot Express utilizes the Raspberry Pi's GPIO (General Purpose Input/Output) pins for interfacing with external devices and sensors. The system leverages GPIO libraries and protocols to communicate with sensors, actuators, and other peripherals connected to the Raspberry Pi, ensuring reliable data exchange and control.

Arduino Interface: FarmBot Express connects to Arduino microcontrollers via a serial or USB interface. FarmBot Express utilizes Arduino compatible firmware and communication protocols to interface with the software, allowing for precise control of motor movements and sensor readings.

Stepper Motors and Rotary Encoders Interface: FarmBot Express uses motor driver interfaces (such as stepper motor driver boards) to control stepper motors and interfaces with rotary encoders for position feedback. Stepper motors are employed for tasks such as gantry movement and tool positioning, while rotary encoders monitor motor position and velocity.

Sensor and Actuator Interfaces: FarmBot Express supports various types of sensors and actuators, utilizing analog, digital, or serial interfaces for connectivity. Sensors such as soil moisture sensors, temperature sensors, and humidity sensors monitor environmental conditions, while actuators like watering heads and seeding mechanisms perform farming tasks. The system accommodates sensor and actuator protocols such as UART and GPIO for seamless data exchange and control, enabling efficient monitoring and automation of farming operations.

1.3.1.4 Software Interfaces

FarmBot OS: FarmBot Os is the custom operating system running on the Raspberry Pi. It provides the foundational software environment for FarmBot's operation, managing hardware resources, executing software components, and facilitating communication between different parts of the system. Several of this interfaces are Raspberry Pi hardware components for hardware control, Farmduino firmware for sending and receiving commands and data, interface with external peripherals and devices connected to the Raspberry Pi.

FarmBot Arduino Firmware (Farmduino): FarmBot Arduino Firmware, also known as Farmduino, is the firmware running on the Arduino microcontroller. It controls Farm.bot's hardware components, including stepper motors, sensors, and peripherals, based on commands received from FarmBot OS.

FarmBot JS Interface: FarmBot JS is a JavaScript library that wraps Farm.bot's authentication and RPC (Remote Procedure Call) instructions. It allows interaction with Farm.bot from web applications.

FarmBot Python Library Interface: FarmBot Python Library is a Python library for interacting with FarmBot. It allows developers to control FarmBot's functionality using Python scripts.

API Interface: The API interface performs HTTP requests to the FarmBot API. It allows external systems or applications to interact with Farm.bot by sending commands and receiving data. Interface with FarmBot's API endpoints for performing CRUD (Create, Read, Update, Delete) operations.

1.3.1.5 Communication Interfaces

Wi-Fi Interface: FarmBot connects to local Wi-Fi networks to enable communication with other devices and systems. Allowing FarmBot to communicate wirelessly within a local network environment.

Ethernet Interface: FarmBot may also support Ethernet connectivity for wired communication within a local network. Ethernet interfaces typically support protocols such as TCP/IP (Transmission Control Protocol/Internet Protocol) for communication over Ethernet networks.

USB Interface: USB interfaces allow FarmBot to connect to external devices such as cameras, sensors, or storage devices for data exchange or control.

Serial Interface: Serial interfaces facilitate communication between Farm.bot and external devices using serial communication protocols. Serial interfaces typically support protocols such as UART for serial communication over cables or connectors.

MQTT: MQTT is a lightweight messaging protocol used for communication between Farm.bot and other devices. MQTT operates over TCP/IP and uses a publish-subscribe model for message exchange, allowing Farm.bot to publish sensor data or subscribe to control commands from MQTT brokers or clients.

HTTP: HTTP is a standard protocol used for communication between Farm.bot and web servers or APIs. HTTP operates over TCP/IP and supports methods such as GET, POST, PUT, and DELETE for exchanging data between Farm.bot and web-based applications or services.

1.3.1.6 Memory Constraints

Memory constraints for the FarmBot project are not a significant concern. The system primarily relies on the Raspberry Pi, which typically utilizes a microSD card for storage and running FarmBot OS. As such, the memory requirements are modest and easily manageable. The microSD card provides sufficient storage capacity for the operating system, software components, and data storage, ensuring the feasibility of the project without imposing memory constraints.

1.3.1.7 Operations

The operations provided by FarmBot are:

User-Initiated Operations:

- Planting seeds
- Watering crops
- Weed
- Harvest crops
- Fertilize seeds
- Mount tools

Unattended Operations:

- Automated watering based on predefined schedules or sensor readings
- Automated harvesting when crops reach maturity
- Sequence Execution

Data Processing Support Functions:

- Logging sensor readings, operation statuses, and other data
- Analyzing data trends and patterns
- Generating reports on farm performance and crop growth

Special Operations:

- Stop at emergency

All details of these operations are in the 3.2 Functions section.

1.3.2 System Functions

Function	**Summary**
Plant seeds	Users can specify type of plant such as spinach, broccoli, beet for FarmBot to plant the seed in the designated place.
Water plants	Users can tell FarmBot to water plants at a spesific area. Also user can specify amount of water day by day.
Harvest crops	Users instruct FarmBot to harvest some spesific crops in the designated place.
Create sequences	Users create sequences which are pre-defined set of actions that FarmBot can execute automatically.
Fertilize seeds	Users fertilize seeds by specifying fertilizer type for a particular place in the garden.
Monitor logs	Users monitor all logs generated by FarmBot system as a result of some event.
Monitor resource usage	Users monitor resource usage of FarmBot system for resource types such as water and fertilizer.
Stop at emergency	User can stop FarmBot system at any emergency condition by clicking E-STOP button.

Table 1.1: System Functions

1.3.3 Stakeholder Characteristics

The system has various stakeholders whose technical experiences and capabilities are ranging from basic users to experts. The stakeholders of the system are **end users (farmers, hobbyist), educational institutions, software developers, government agencies.**

- **End Users:** End users are the primary beneficiaries and users of the FarmBot system. Farmers utilize FarmBot to automate tasks such as planting, watering, and weeding on their farms. Hobbyists may also use FarmBot for personal gardening projects or educational purposes, exploring its capabilities in a smaller-scale setting. End users require user-friendly interfaces, reliable hardware, and accessible support for troubleshooting and maintenance to effectively integrate FarmBot into their agricultural practices or personal projects.

- **Educational Institutions:** Educational institutions, including universities and high schools, view FarmBot as a valuable tool for teaching subjects such as agriculture, and robotics. By integrating FarmBot into their curriculum, these institutions provide students with hands-on learning experiences and research opportunities. Additionally, FarmBot serves as a platform for exploring concepts such as automation, sustainability, and food production.

- **Government Agencies:** Government agencies such as the Ministry of Agriculture and Forestry are interested in sustainable agriculture and advancing technological solutions to address challenges in the agricultural sector. They recognize the potential of FarmBot to increase productivity, preserve resources, and improve food efficiency. Additionally, organizations like NASA may explore FarmBot technology for applications in space agriculture, supporting long duration space missions.

- **Software Developers:** Developers contribute to the FarmBot project as part of an open source community, collaborating to improve and expand the capabilities of the system. Developers from diverse backgrounds bring expertise in software development, robotics, and agriculture to the FarmBot ecosystem. They work on tasks such as developing control software, designing hardware components, and improving user interfaces.

1.3.4 Limitations

- **Regulatory Policies:** The system holds mail address and password information about the user. The system should encrypt these user information and also, system should comply with regulations about usage of water and pesticide against harmful plants.

- **Hardware Limitations:** The FarmBot system uses several hardware components, to ensure flawless work. These hardware components are sensors (light, moisture), camera, Arduino, and Raspberry Pi. There are some limitations for precision and sensitivity of sensors, resolution and processing capabilities of the camera, memory and processing constraints of Arduino and Raspberry Pi, as well as storage capacity and reliability of the micro SD card.

- **Interfaces to other applications:** FarmBot currently does not have interfaces to other applications, but there is potential implementation in future.

- **Parallel operation:** FarmBot is not able to do parallel operations since it has just one Arduino and necessary hardware components. Design of the system does not allow parallel operations.

- **Audit functions:** FarmBot's audit functions limited with tracking and logging farm activities. There might be potential logging constraints on data collected from sensors and system events.

- **Control functions:** FarmBot's control functions are limited by the capabilities of hardware and software architecture. Current FarmBot implementation uses a Raspberry Pi for controls. Control functions limitations is same as Raspberry Pi's limitations.

- **Higher-order language requirements:** Currently, system supports languages commonly used in microcontroller environments, such as C++ for Arduino based development, Elixir for FarmBot_OS and Python for communication with Raspberry Pi. System also uses Ruby and TypeScript for WebApp.

- **Signal handshake protocols:** Mobile and desktop FarmBot apps utilize the HTTPS protocol for sending and receiving information from FarmBot server. Additionally FarmBot uses alternative protocols such as MQTT to enhance reliability.

- **Quality requirements:** FarmBot's hardware should be durable against harsh agricultural conditions. Also software should be stable to determine the overall quality of the system. Both software and hardware conditions determines the quality of system.

- **Criticality of the application:** If FarmBot does not work properly, it could have significant implications for agricultural operations of user. Any malfunction or downtime may lead financial loss or crop loss.

- **Safety and security considerations:** Since FarmBot system does not handle any personal data and operates as an open source project, it does not have much security and safety constraints. Also basic safety protocols should still be observed to minimize potential risks.

- **Physical/mental considerations:** FarmBot system designed to prioritize user friendliness and it does not require physical activity to interact with farm environment, so it ensures accessibility for individuals with disabilities.

- **Limitations that are sourced from other systems:** FarmBot may encounter limitations in particularly meeting real-time requirements for data coming from external entities such as openfarm.cc.

1.4 Definitions

Term	**Definition**
API	Application Programming Interface
CRUD	Create, Read, Update, Delete
GPIO	General Purpose Input/Output
HTTP	Hypertext Transfer Protocol
MQTT	Message Queuing Telemetry Transport
RPC	Remote Procedure Call
TCP/IP	Transmission Control Protocol/Internet Protocol
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus

Table 1.2: Definitions

2. References

This document is written with respect to the IEEE 29148-2018 standards:

29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering

3. Specific Requirements

3.1 External Interfaces

The following class diagram represents the relationship between interfaces and their functionalities. For an explanation of the interfaces, please refer to section (1.3.1.2).

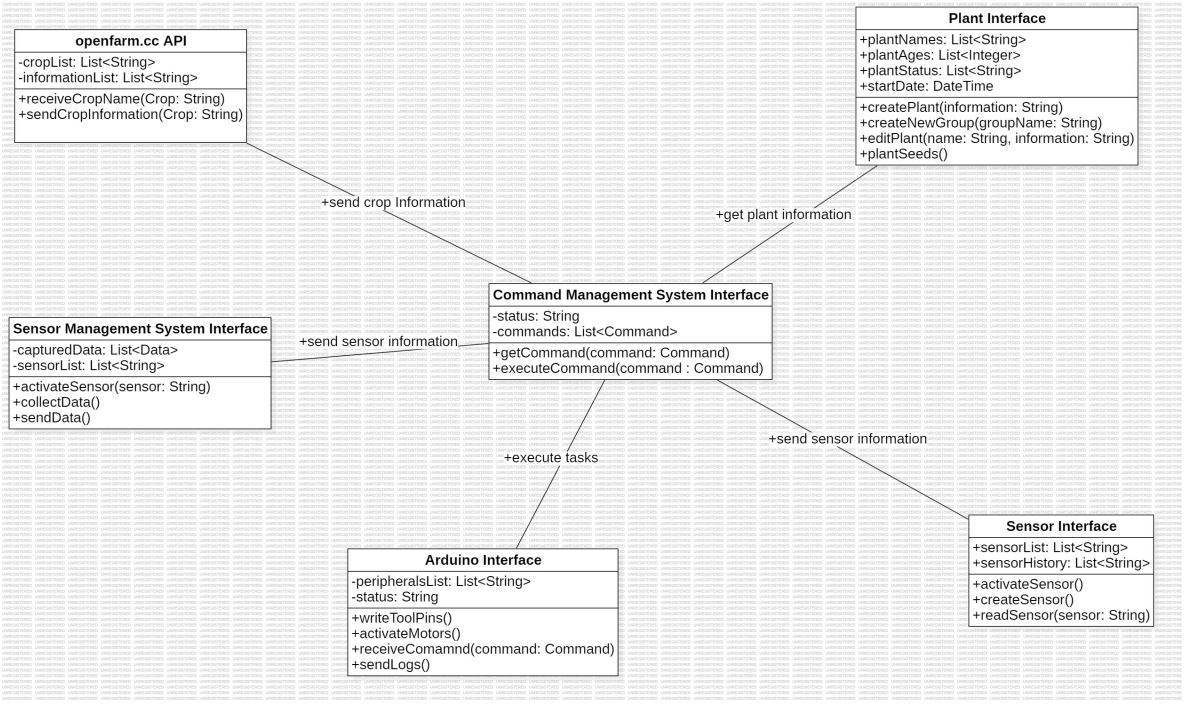

Figure 3.1: External Interfaces Class Diagram

3.2 Functions

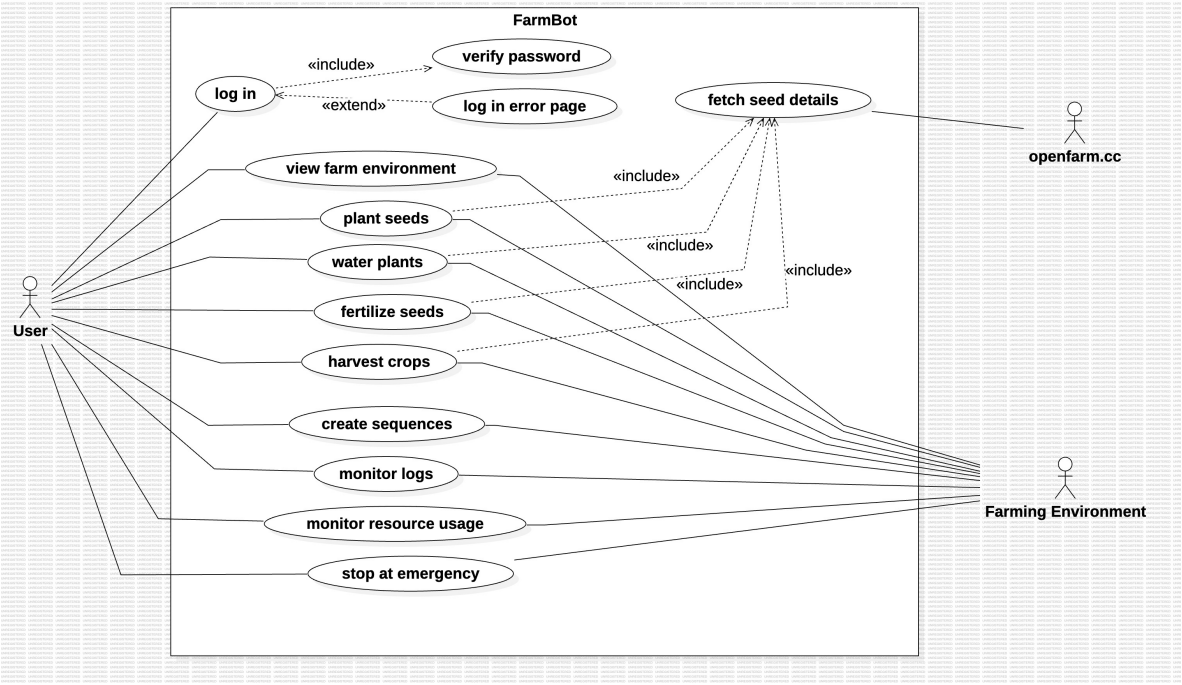

Figure 3.2: Use Case Diagram

Use Case Name	Login
Actors	User
Description	User logs into the FarmBot system using their credentials.
Data	User credentials (username, password)
Preconditions	The FarmBot system is operational and accessible via a web interface or mobile app
Stimulus	User initiates the login process by accessing the FarmBot login page
Basic Flow	Step 1: User enters their username and password. Step 2: System verifies the user's credentials. Step 3: If the credentials are valid, the system grants access to the user's account.
Alternative Flow #1	Step 1: User enters their username and password. Step 2: System verifies the user's credentials. Step 3: If the entered username or password is invalid, the system displays an error message indicating that the username or password is incorrect
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the login process may fail, and the user will be unable to access their account
Postconditions	User is logged into the FarmBot system and has access to their account dashboard or main interface

Table 3.1: Login

Use Case Name	Verify Password
Actors	User
Description	This use case describes the process by which the FarmBot system verifies the password provided by the user during the login process.
Data	User credentials (username, password)
Preconditions	The FarmBot login process has been initiated.
Stimulus	User submits their password for verification.
Basic Flow	Step 1: User enters their password during the login process. Step 2: The system compares the user-provided password with the stored password. Step 3: If the passwords match, the system grants access to the user's account.
Alternative Flow #1	Step 3a: If the entered username or password is invalid, the system displays an error message indicating that the username or password is incorrect.
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the verify password process may fail, and An error message is displayed indicating the issue.
Postconditions	User gains access to their FarmBot account.

Table 3.2: Verify Password

Use Case Name	Log in Error Page
Actors	User
Description	The FarmBot system presents an error page to the user in case of login failure, providing feedback and guidance for resolving the issue.
Data	Error message, login credentials.
Preconditions	The FarmBot login process has been initiated.
Stimulus	User attempts to log in with incorrect credentials.
Basic Flow	Step 1: User enters their password during the login process. Step 2: The system compares the user-provided password with the stored password. Step 3: If the credentials are incorrect. The system redirects the user to an error page specifically designed to handle login failures.
Alternative Flow #1	-
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the Log in Error Page process may fail, and An error message is displayed indicating the issue.
Postconditions	The user is presented with an error page containing feedback and guidance for resolving the login failure.

Table 3.3: Log in Error Page

Use Case Name	View farm environment
Actors	User, Farming Environment
Description	User views the current environment and conditions of their farm, including temperature, humidity, soil moisture, and other relevant data.
Data	Environmental data such as, temperature, humidity, soil moisture, user preferences.
Preconditions	The FarmBot system is operational and accessible via a web interface or mobile app
Stimulus	User initiates the request to view the farm environment through the FarmBot interface.
Basic Flow	Step 1: User navigates to the "Map" section of the FarmBot dashboard. Step 2: The system retrieves real-time environmental data from connected sensors, including temperature, humidity, and soil moisture levels. Step 3: The system displays the collected environmental data in a user-friendly format on the dashboard interface. Step 4: User may choose to customize the display or view historical data based on preferences.
Alternative Flow #1	-
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the login process may fail, and the user will be unable view farming environment
Postconditions	User has access to real-time and/or historical environmental data of their farm.

Table 3.4: View Farm Environment

Use Case Name	Plant Seeds
Actors	User, Farming Environment
Description	User instructs the FarmBot system to plant seeds in the designated areas of the garden bed.
Data	Seed types, planting schedule, planting depth, planting density.
Preconditions	The FarmBot system is operational, the garden bed configuration is set up, and seeds are available for planting.
Stimulus	User initiates the request to plant seeds through the FarmBot interface.
Basic Flow	Step 1: User navigates to the "Plants" section. Step 2: User selects the type of seeds they want to plant from a list of available options. Step 3: User specifies the planting schedule, planting depth, and planting density for the selected seeds. Step 4: The system retrieves the current configuration of the garden bed and identifies suitable planting locations based on the specified parameters
Alternative Flow #1	Step 3a: If the system detects any issues with the designated planting locations (e.g., soil compaction, existing plant obstructions), FarmBot alerts the user and suggests alternative planting locations.
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the login process may fail, and the user will be unable view farming environment
Postconditions	Seeds are successfully planted in the designated areas of the garden bed. System updates planting records and schedules for future reference.

Table 3.5: Plant Seeds

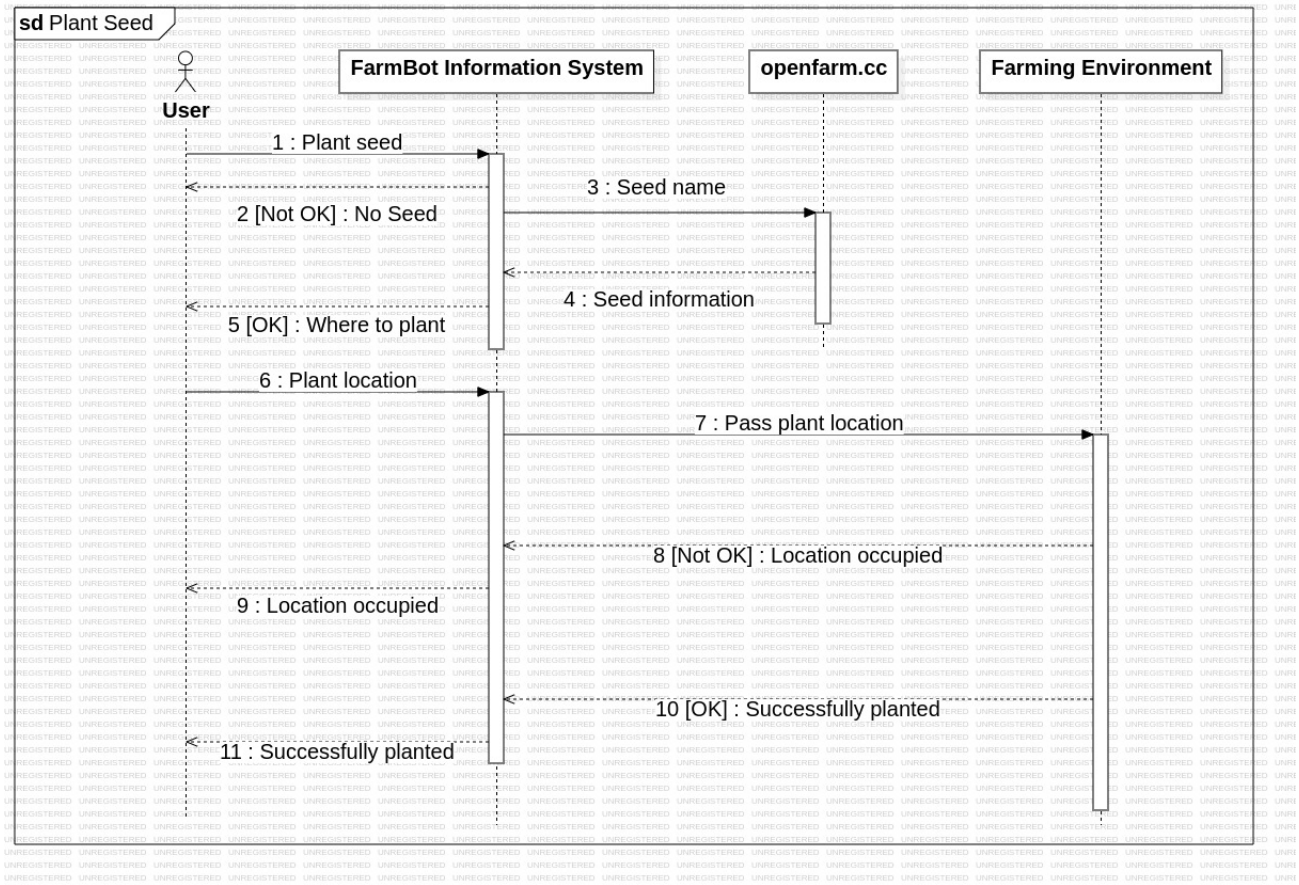

Figure 3.3: Plant Seed Sequence Diagram

Use Case Name	Water Plants
Actors	User, Farming Environment
Description	User instructs the FarmBot system to water plants in the designated areas of the garden bed.
Data	Watering schedule, watering duration, watering intensity.
Preconditions	The FarmBot system is operational, the garden bed configuration is set up, and plants require watering.
Stimulus	User initiates the request to water plants through the FarmBot interface.
Basic Flow	Step 1: User navigates to the "Watering" section. Step 2: User specifies the watering schedule, duration, and intensity for watering plants. Step 3: The system retrieves the current configuration of the garden bed and identifies plants requiring watering based on the specified parameters. Step 4: FarmBot moves to the designated watering locations and initiates the watering process.
Alternative Flow #1	Step 3a: If the system detects any issues with the designated watering locations (e.g. insufficient water supply), FarmBot alerts the user and suggests corrective actions.
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the watering process may fail, and the user will be unable to water plants.
Postconditions	Plants are successfully watered in the designated areas of the garden bed. System updates watering records and schedules for future reference.

Table 3.6: Water Plants

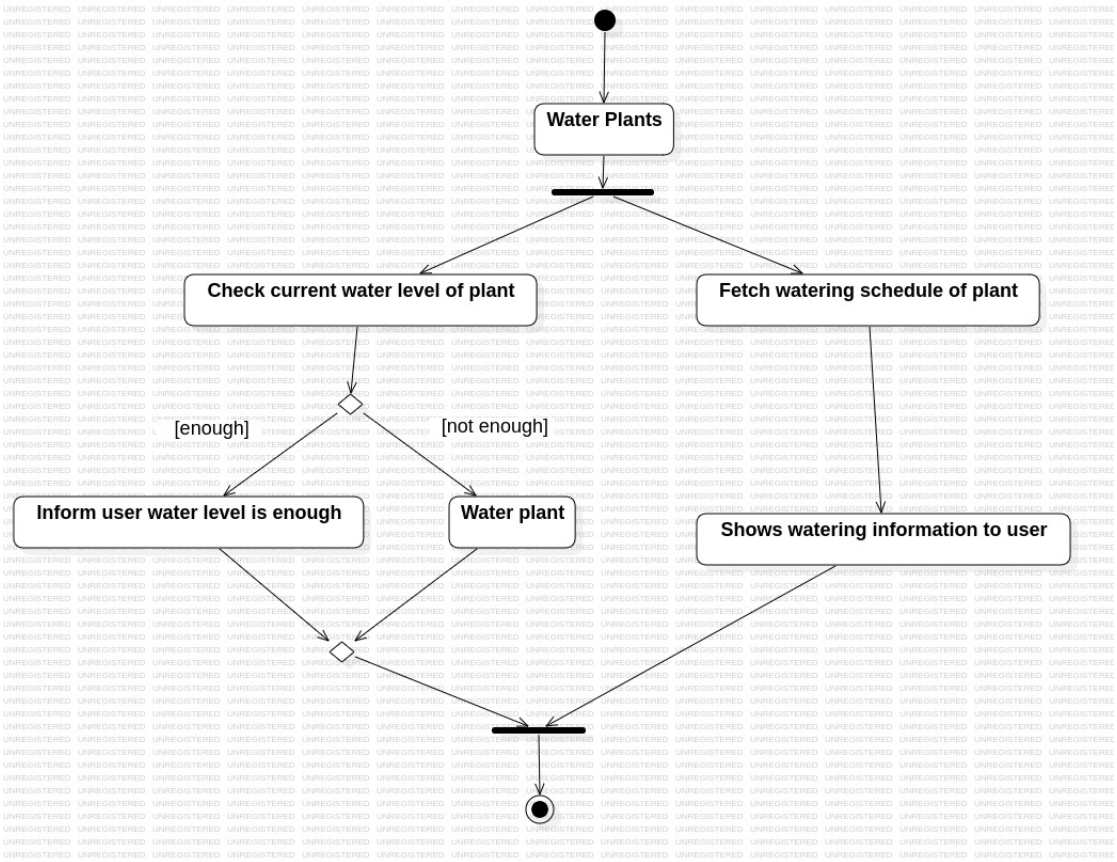

Figure 3.4: Water Plants Activity Diagram

Use Case Name	Fertilize Seeds
Actors	User, Farming Environment
Description	User instructs the FarmBot system to fertilize seeds in the designated areas of the garden bed.
Data	Fertilizer type, fertilizing schedule, fertilizing quantity.
Preconditions	The FarmBot system is operational, the garden bed configuration is set up, and seeds are planted and require fertilization.
Stimulus	User initiates the request to fertilize seeds through the FarmBot interface.
Basic Flow	Step 1: User navigates to the "Fertilization" section. Step 2: User specifies the type of fertilizer, fertilizing schedule, and quantity for fertilizing seeds. Step 3: The system retrieves the current configuration of the garden bed and identifies seeds requiring fertilization based on the specified parameters. Step 4: FarmBot moves to the designated planting locations and initiates the fertilization process.
Alternative Flow #1	Step 3a: If the system detects any issues with the designated fertilization locations (e.g. inappropriate fertilizer type), FarmBot alerts the user and suggests corrective actions.
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the fertilization process may fail, and the user will be unable to fertilize seeds.
Postconditions	Seeds are successfully fertilized in the designated areas of the garden bed. System updates fertilization records and schedules for future reference.

Table 3.7: Fertilize Seeds

Use Case Name	Harvest Crops
Actors	User, Farming Environment
Description	User instructs the FarmBot system to harvest mature crops from the designated areas of the garden bed.
Data	Harvesting schedule, crop types, harvesting method.
Preconditions	The FarmBot system is operational, the garden bed configuration is set up, and crops are mature and ready for harvesting.
Stimulus	User initiates the request to harvest crops through the FarmBot interface.
Basic Flow	Step 1: User navigates to the "Harvesting" section. Step 2: User specifies the crop types and harvesting schedule for the crops to be harvested. Step 3: The system retrieves the current configuration of the garden bed and identifies mature crops based on the specified parameters. Step 4: FarmBot moves to the designated harvesting locations and initiates the harvesting process.
Alternative Flow #1	Step 3a: If the system detects crops not ready for harvesting, harvesting operations stop, system returns homepage.
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the harvesting process may fail, and the user will be unable to harvest crops.
Postconditions	Crops are successfully harvested from the designated areas of the garden bed. System updates harvesting records and schedules for future reference.

Table 3.8: Harvest Crops

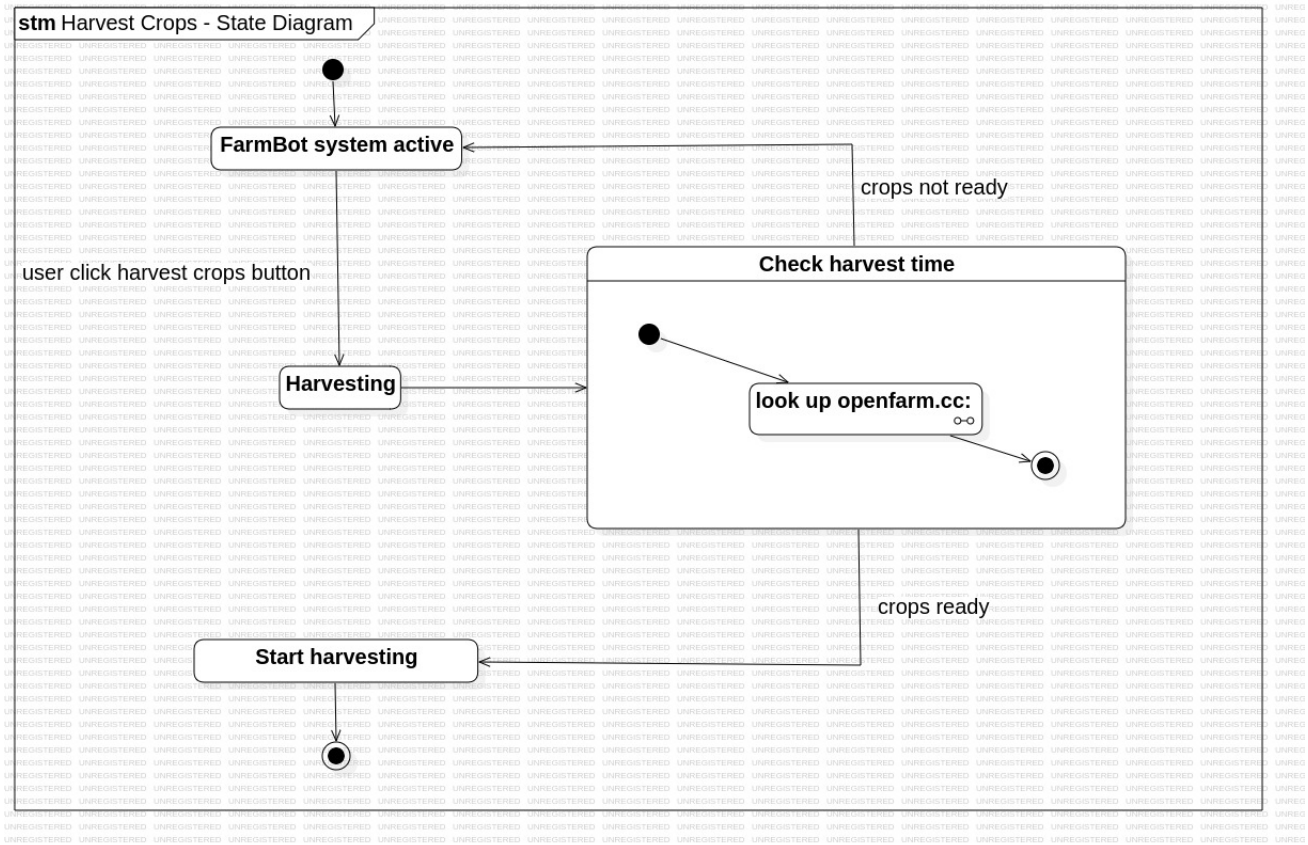

Figure 3.5: Harvest Crops State Diagram

Use Case Name	Fetch Seeds Details
Actors	User, openfarm.cc
Description	The FarmBot system retrieves detailed information about a specific seed variety from the openfarm.cc database, providing users with essential details for planting and cultivation.
Data	Seed variety name.
Preconditions	The FarmBot system is operational and connected to the internet.
Stimulus	User initiates a request to fetch seed details through the FarmBot interface.
Basic Flow	Step 1: User selects a specific seed variety for which they want to fetch details. Step 2: The system sends a request to the openfarm.cc database, providing the name of the selected seed variety. Step 3: The system retrieves the current configuration of the garden bed and identifies mature crops based on the specified parameters. Step 4: The system receives the seed details from openfarm.cc and displays them to the user on the FarmBot interface.
Alternative Flow #1	-
Exception Flow	If the FarmBot system is inaccessible or experiencing technical issues, the process may fail, and the user will be unable to Fetch Seeds Details.
Postconditions	User is presented with detailed information about the selected seed variety fetched from the openfarm.cc database.

Table 3.9: Fetch Seeds Details

Use Case Name	Create Sequences
Actors	User
Description	User creates sequences of actions for the Farmbot to execute.
Data	Actions and parameters of sequences
Preconditions	User should be logged in and connected to the Farmbot.
Stimulus	User clicks new sequence button.
Basic Flow	Step 1: User spesifies desired actions and parameters for new sequence. Step 2: User clicks save button.
Alternative Flow #1	Step 1: User clicks existing sequence to modify it. Step 2: User makes neccessary changes on actions and variables. Step 3: User clicks save button.
Exception Flow	If any error occurs while creating a new sequence, display an error message and let user fix the problem.
Postconditions	New sequence is created and saved in the system.

Table 3.10: Create Sequences

Use Case Name	Monitor Logs
Actors	User
Description	User monitors the logs generated by the Farmbot system to track events, errors, or other relevant information.
Data	Log timestamp and event description data.
Preconditions	User should be logged in and connected to the Farmbot.
Stimulus	User clicks to logs button.
Basic Flow	Step 1: System displays a list of latest logs with timestamps and descriptions of events. Step 2: User reviews the listed logs for relevant information.
Alternative Flow #1	Step 1: User applies a filter to get some specific logs. Step 2: System displays the filtered logs.
Exception Flow	If any error occurs while trying to see logs, display and error message and let the user try again.
Postconditions	User has reviewed the desired logs.

Table 3.11: Monitor Logs

Use Case Name	Monitor Resource Usage
Actors	User
Description	User monitors the usage of resources such as water, fertilizer, and electricity by the Farmbot system.
Data	Log timestamp and event description data.
Preconditions	User should be logged in and connected to the Farmbot.
Stimulus	User clicks to logs button.
Basic Flow	Step 1: System displays a list of latest logs with timestamps and descriptions of events. Step 2: User reviews the listed logs for relevant information.
Alternative Flow #1	Step 1: User applies a filter to get some spesific logs. Step 2: System displays the filtered logs.
Exception Flow	If any error occurs while trying to see logs, display and error message and let the user try again.
Postconditions	User has reviewed the desired logs.

Table 3.12: Monitor Resource Usage

Use Case Name	Stop at Emergency
Actors	User
Description	User can use emergency stop to halt movements and turn off peripherals of FarmBot.
Data	-
Preconditions	User should be logged in and connected to the Farmbot.
Stimulus	User clicks to E-STOP button.
Basic Flow	Step 1: FarmBot stops all movements. Step 2: FarmBot goes into the locked state.
Alternative Flow #1	-
Exception Flow	If any problem occurs while executing emergency stop, notify the user and give guidance to user to resolve the issue.
Postconditions	FarmBot has stopped and user need to press the button again to reactive the system.

Table 3.13: Stop at Emergency

3.3 Logical Database Requirements

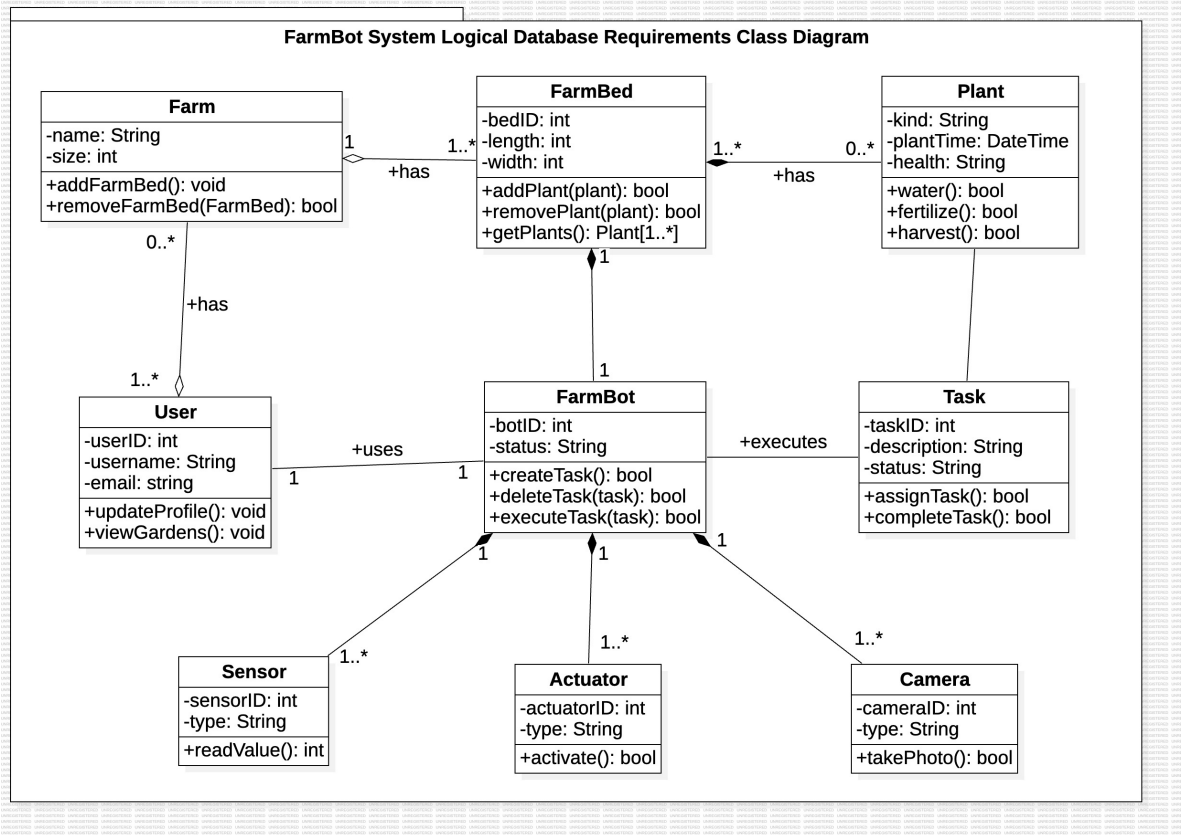

Figure 3.6: Logical Database Class Diagram

- addFarmBed operation allows users to expand their farming area by adding new garden beds, increasing the capacity for plant cultivation.
- Users can remove a garden bed from the farm with removeFarmBed operation, either to reconfigure the layout or reduce the farming area.
- User populates garden beds with desired plants using the addPlant(plant) operation, facilitating precise control over plant placement and variety within the farm.
- User can selectively remove plants from garden beds using the removePlant(plant)

operation, allowing for replanting or adjustment of the bed's composition to optimize plant growth.

- User retrieves a comprehensive list of plants currently growing in a specific garden bed with the getPlants() operation to monitor and manage of plant growth.
- User ensures optimal watering for plants by triggering the watering mechanism as needed with the water() operation, promoting healthy growth and development.
- User supplies essential nutrients to plants by applying fertilizer using the fertilize() operation, supporting growth and crop yield.
- User collects mature crops with the harvesting process with the harvest() operation.
- User maintains accurate personal details, such as username and email address, using the updateProfile() operation, ensuring effective communication and user management within the system.
- User accesses and visualizes the gardens they own or manage with the viewGardens() operation to strategic planning of farming activities.
- User delegates specific farm tasks to FarmBots for automated execution using the assignTask() operation, optimizing farm management efficiency and reducing manual works.
- User tracks and manages task progress by marking assigned tasks as completed once FarmBots finish executing them with the completeTask() operation, ensuring task management.
- FarmBots execute assigned tasks autonomously, such as planting, watering, or harvesting operations with the ExecuteTask(task).
- Sensors gather environmental data such as soil moisture or temperature with the readValue() operation, providing valuable insights farm management decisions.

- A user triggers specific actions, such as watering plants or adjusting equipment positions, by activating actuators with the activate() operation, with this function user has precise control over farm operations and equipment.

3.4 Design Constraints

- The FarmBot project must comply with relevant agricultural regulations and standards setby local, regional, or national authorities.
- It should follow agricultural machinery and equipment safety standards.
- The design of the FarmBot system must stick to to environmental regulations about pesticide use, water use, and soil management.
- Use of environmentally friendly materials and methods should be prioritized to minimize ecological impact.
- The system must follow to data privacy regulations and ensure the security of user data collected by sensors, cameras, and other monitoring devices.

3.5 System Quality Attributes

a) **Reliability**

- Software components responsible for task execution should have built in fault tolerance mechanisms to handle unexpected situations.

b) **Availability**

- The system should be available for use whenever users require farm management tasks to be performed.
- FarmBot should have high uptime, ensuring that tasks can be assigned and executed.

c) **Security**

- Access to the FarmBot system should be restricted to authorized users only.
- Data transmitted between FarmBots and control systems should be encrypted.
- Regular security audits and updates should be conducted to identify and address potential vulnerabilities in the system.

d) **Maintainability**

- The software well documented to facilitate ease of maintenance and future enhancements.
- Automated testing and deployment pipelines should be created to the process of deploying updates and bug fixes.

e) **Portability**

- Configuration settings and user preferences should be easily transferable between different instances of the FarmBot system to support scalability and deployment in different farming setups.

3.6 Supporting Information

FarmBot is an open source project, contributions from individuals worldwide to collaborate on its development and improvement. This open approach encourage innovation and community engagement. This approach allowing hobbyist, farmers and developers to contribute their expertise and ideas to the project. Additionally, FarmBot provides openly available 3D CAD models, this enables anyone to access and download the designs for FarmBot hardware components. With using these models, individuals can utilize 3D priters to manufacture their own FarmBot components. This open philosophy empowers users to customize and iterate upon the design, and all of this efforts lead further innovation and sustainable farming.

4. Suggestions to Improve The Existing System

- **OpenAI API Integration:** By integrating the OpenAI API into the FarmBot system, users can receive personalized farming suggestions and tips. The AI-powered suggestions can cover planting schedules, pest control and harvesting times.

- **Gmail API Integration:** Utilizing the Gmail API, the FarmBot system can automatically send users weekly or monthly summaries of their farming activities. These summaries can include metrics such as crop growth progress, water and energy usage, pest alerts, and harvest yields. By receiving regular updates via email, users can stay informed about the status of their farm without having to actively monitor the system.

- **Gamification:** Introducing gamification elements such as medals, streak points, or badges within the FarmBot system enhances user motivation and engagement, especially for educational purposes. Users can earn rewards or achievements based on their farming accomplishments, such as successfully growing a certain number of crops, maintaining a consistent watering schedule, or achieving high crop yields.

- **Accessibilty Features:** By integrating voice command functionality and screen reader compatibility, the FarmBot system ensures accessible for users with disabilities.

4.1 System Perspective

FarmBot's functionality can be enhanced with integrations like the OpenAI API and the Gmail API. By integrating the OpenAI API, FarmBot could offer personalized and beneficial suggestions to users' farming needs. This feature could provide helpful guidance on crop selection, planting techniques and pest control strategies. Furthermore, incorporating the Gmail API would enable FarmBot to automatically send users weekly and monthly farm summary updates via email. These updates, containing detailed reports on farm activities and personalized recommendations. With this updates users stay informed from anywhere. Integrating these APIs would undoubtedly enhance user experience.

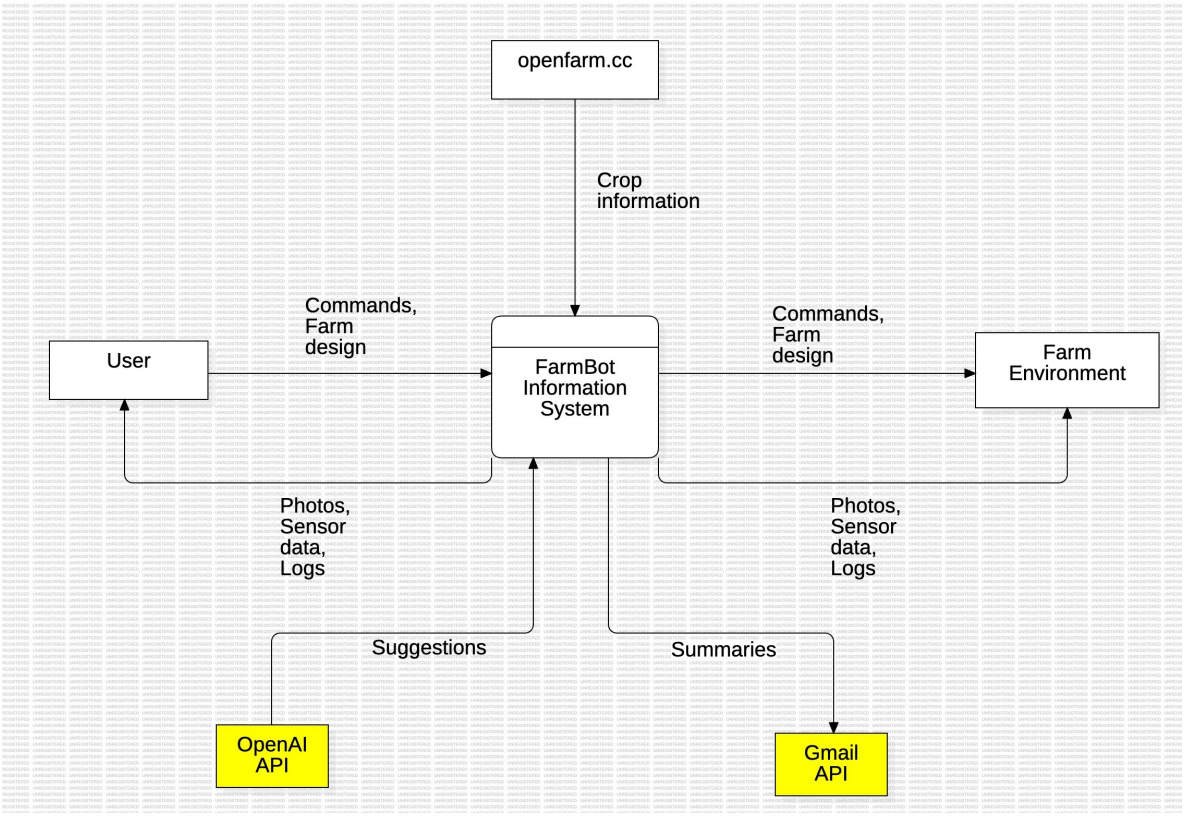

Figure 4.1: Context Diagram (Suggested)

4.2 External Interfaces

- **OpenAI API:** Through this integration, FarmBot gains access to OpenAI's AI models. Users can interact with FarmBot using natural language queries, commands, and requests. This integration give users to personalized recommendations, obtain insights, and receive assistance on various farming tasks, such as crop selection, planting techniques, pest management, and soil health optimization.
- **Gmail API:** By using the Gmail API, FarmBot automates the process of sending weekly and monthly farm summary updates directly to user. These updates contain reports on farm activities, upcoming tasks, and personalized recommendations, empowering users to stay informed and manage their farms effectively from anywhere.

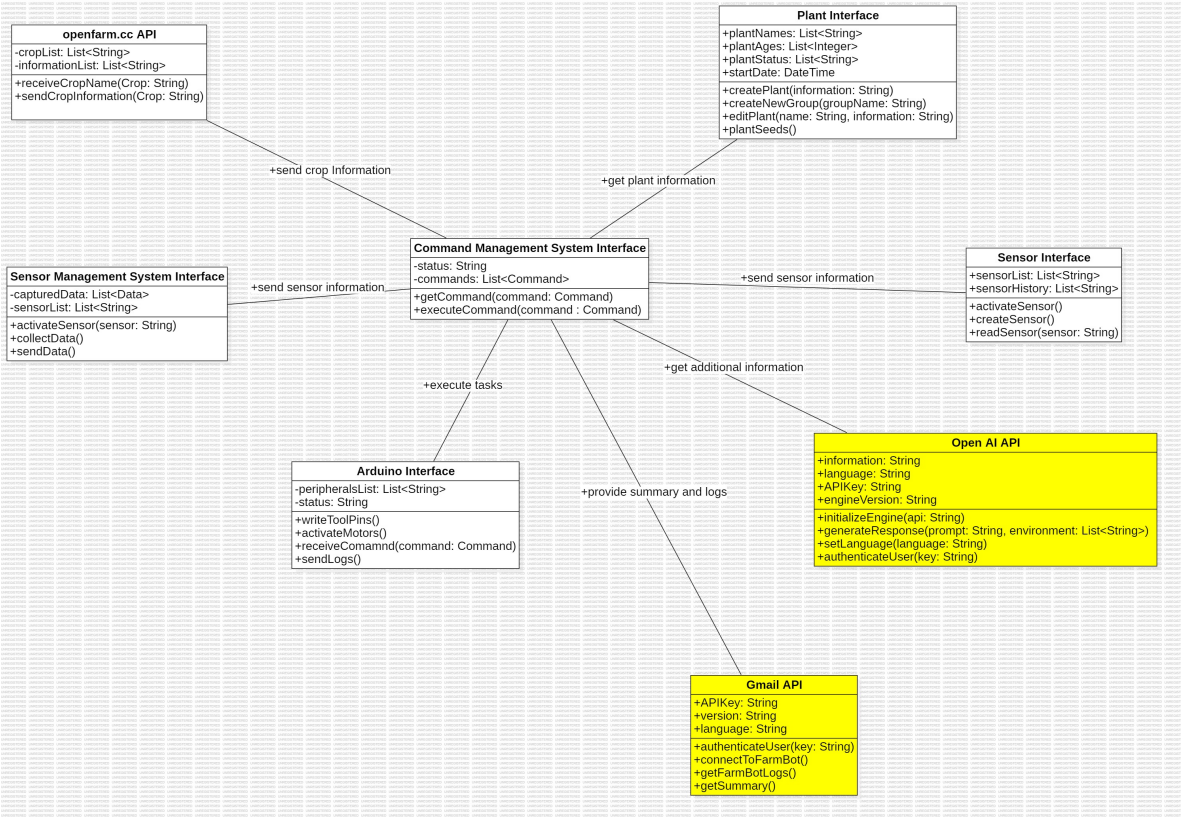

Figure 4.2: External Interfaces Class Diagram (Suggested)

4.3 Functions

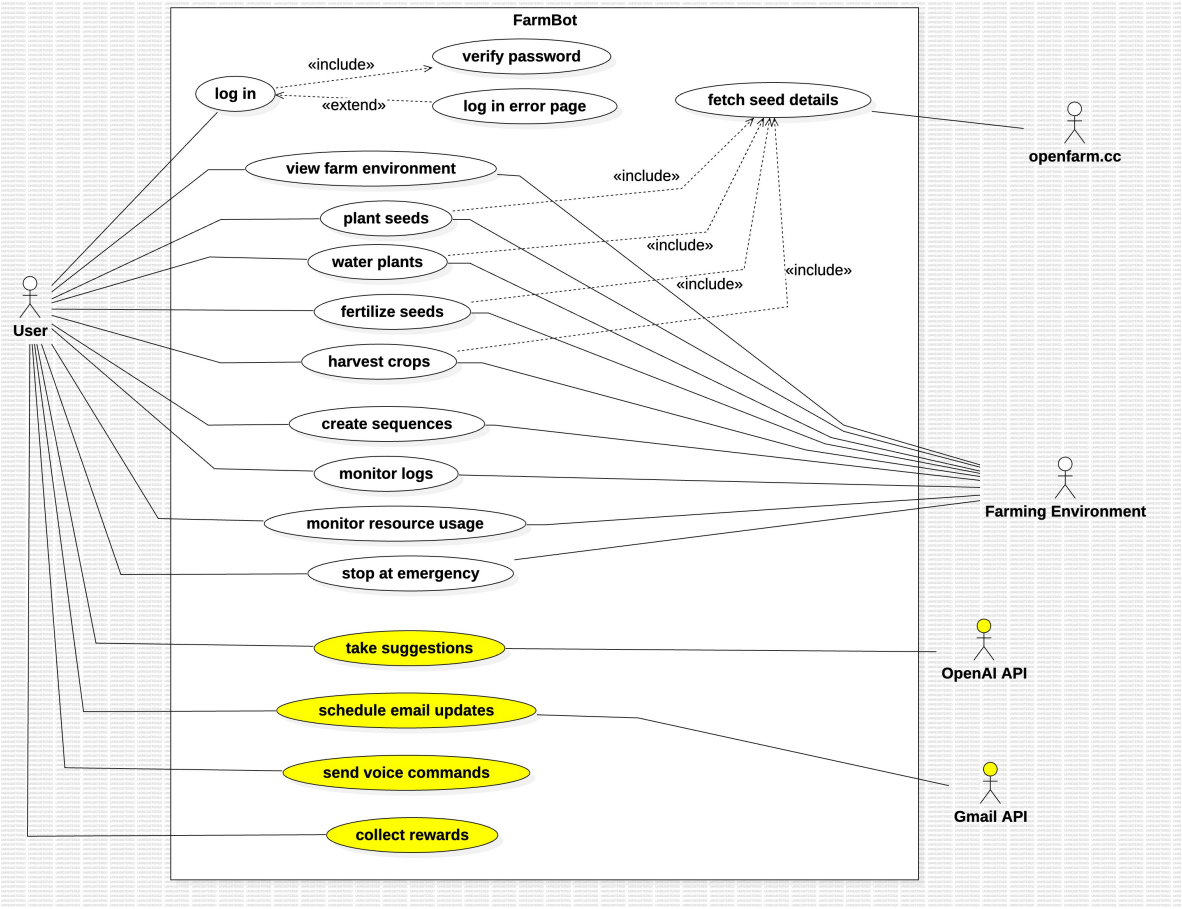

Figure 4.3: Use Case Diagram (Suggested)

Use Case Name	Take Suggestion
Actors	User, OpenAI API
Description	Users interact with the FarmBot system to receive personalized farming suggestions generated by the OpenAI API.
Data	User preferences, farming context such as plant type, location, weather conditions
Preconditions	User must have access to the FarmBot system and a stable internet connection. OpenAI API integration with the FarmBot system must be functional.
Stimulus	User requests farming suggestions from the FarmBot system.
Basic Flow	Step 1: User accesses the FarmBot system interface. Step 2: User inputs request for farming suggestions. Step 3: FarmBot system sends request to the ChatGPT API. Step 4: ChatGPT API processes request and generates personalized farming suggestions. Step 5: FarmBot system presents suggestions to the user.
Alternative Flow #1	-
Exception Flow	If the OpenAI API is unavailable the FarmBot system notifies the user of the inability to provide suggestions.
Postconditions	User receives personalized farming suggestions from the FarmBot system.

Table 4.1: Take Suggestion

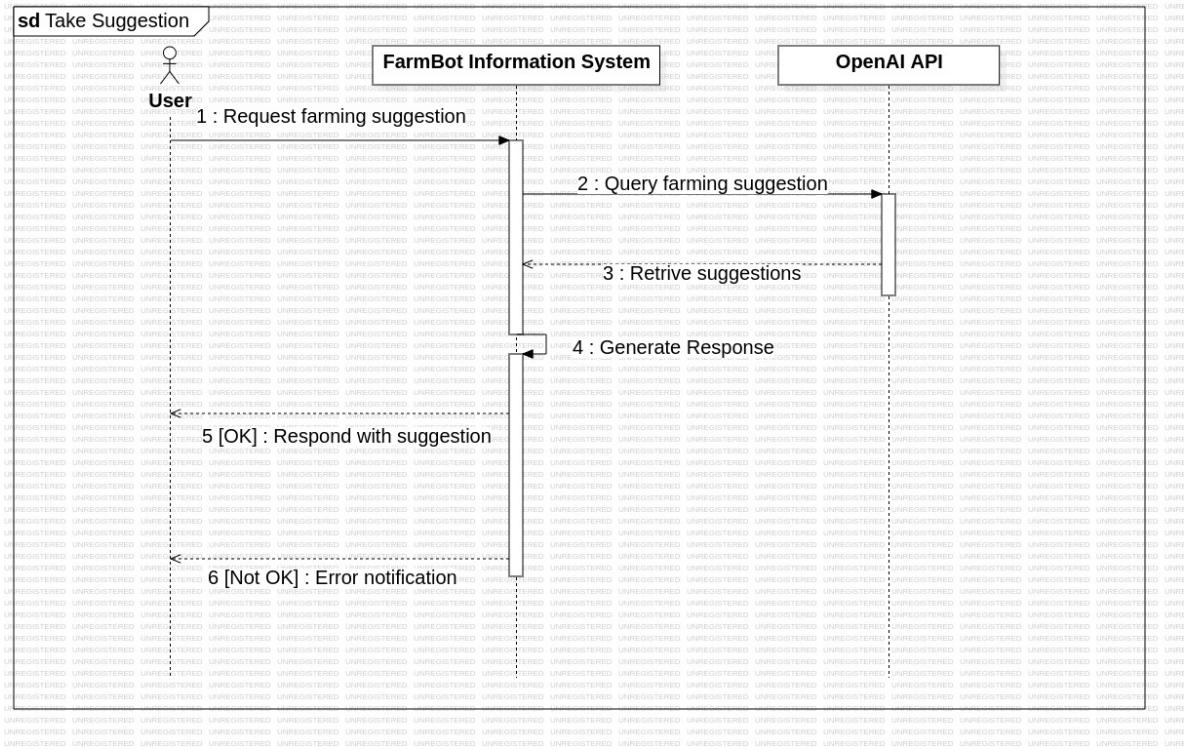

Figure 4.4: Take Suggestion Sequence Diagram (Suggested)

Use Case Name	Schedule Email Updates
Actors	User, Gmail API
Description	Users schedule regular email updates through the FarmBot system using the Gmail API to receive summaries of their farming activities.
Data	User email preferences, farming metrics (e.g., crop growth progress, water usage, pest alerts)
Preconditions	User must have access to the FarmBot system and a stable internet connection. Gmail API integration with the FarmBot system must be functional.
Stimulus	User requests to schedule email updates for farming summaries.
Basic Flow	Step 1: User accesses the settings or preferences section of the FarmBot system. Step 2: User selects the option to schedule email updates. Step 3: User specifies the frequency (weekly or monthly) and timing for receiving email updates. Step 4: FarmBot system saves user preferences and schedules email updates using the Gmail API. Step 5: Gmail API sends farming summaries to the user's email address according to the scheduled frequency
Alternative Flow #1	-
Exception Flow	If the Gmail API is unavailable the FarmBot system notifies the user, schedule task did not completed.
Postconditions	User receives scheduled email updates containing summaries of their farming activities.

Table 4.2: Schedule Email Updates

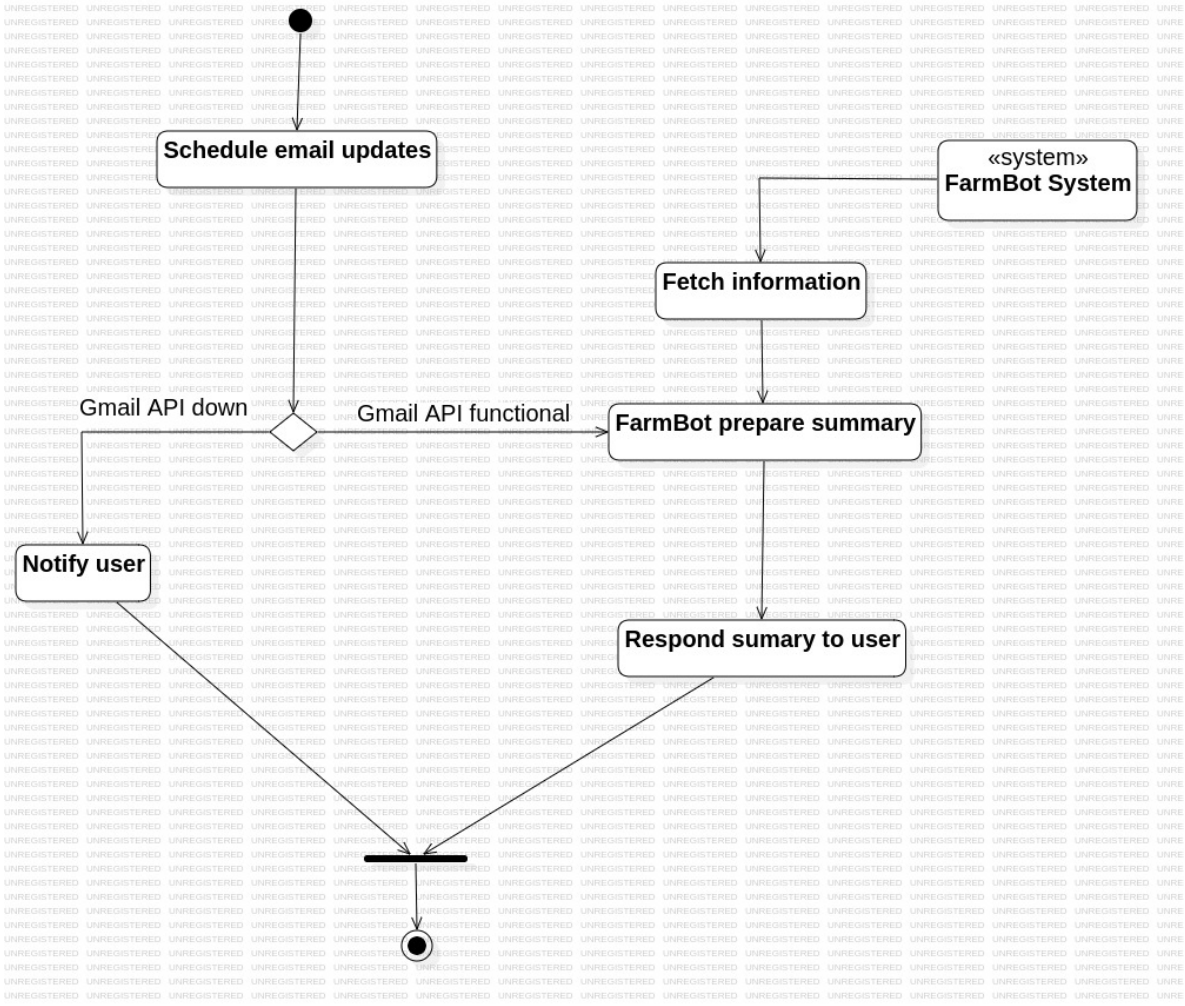

Figure 4.5: Schedule Email Updates Activity Diagram (Suggested)

Use Case Name	Send Voice Commands
Actors	User
Description	Users utilize voice commands to interact with the FarmBot system.
Data	Voice commands.
Preconditions	User must have access to a device with a microphone and the FarmBot system must support voice command functionality.
Stimulus	User speaks voice commands to control the FarmBot system.
Basic Flow	Step 1: User activates the voice command feature on the FarmBot system. Step 2: User speaks a voice command (e.g., "Water plants," "Check soil moisture"). Step 3: FarmBot system processes the voice command. Step 4: FarmBot system executes the corresponding action based on the recognized command.
Alternative Flow #1	-
Exception Flow	If the FarmBot system fails to recognize the voice command due to background noise or some other external factor, it prompts the user to repeat the command.
Postconditions	FarmBot system responds to the user's voice commands and performs the requested actions.

Table 4.3: Send Voice Commands

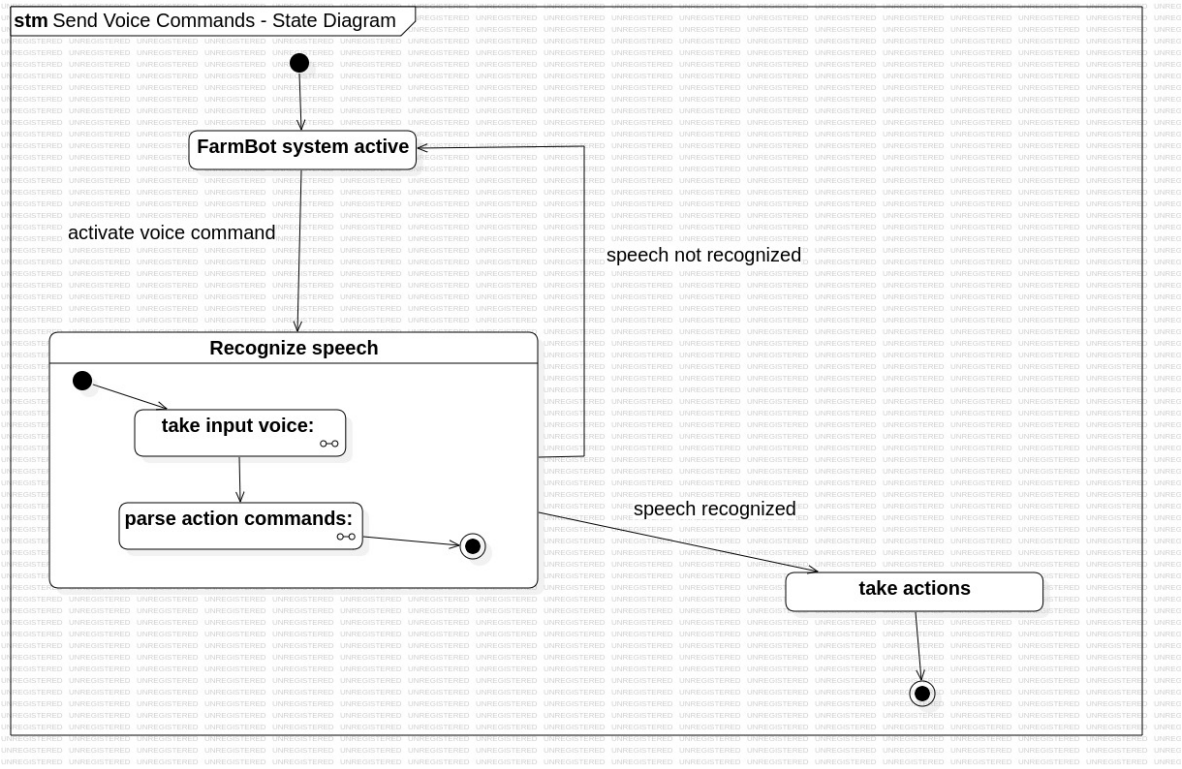

Figure 4.6: Send Voice Commands State Diagram (Suggested)

Use Case Name	Collect Rewards
Actors	User
Description	The "Collect Rewards" use case involves the implementation of gamification elements within the FarmBot system to enhance user motivation and engagement. By integrating features such as medals, streak points, or badges, users are incentivized to actively participate in farming activities
Data	User rewards, farming accomplishments.
Preconditions	User must have completed farming tasks or achieved certain milestones within the FarmBot system to be eligible for rewards.
Stimulus	User must have completed farming tasks or achieved certain milestones within the FarmBot system to be eligible for rewards.
Basic Flow	Step 1: User accomplishes a farming task or achieves a milestone within the FarmBot system. Step 2: FarmBot system recognizes the user's accomplishment. Step 3: FarmBot system awards the user with a reward, such as a medal, streak points, or badge. Step 4: User acknowledges and collects the reward.
Alternative Flow #1	-
Exception Flow	If the FarmBot system fails to recognize the user's accomplishment or encounters errors while awarding the reward, it notifies the user and attempts to resolve the issue.
Postconditions	User successfully collects the reward for their farming accomplishment within the FarmBot system.

Table 4.4: Collect Rewards

4.4 Logical Database Requirements

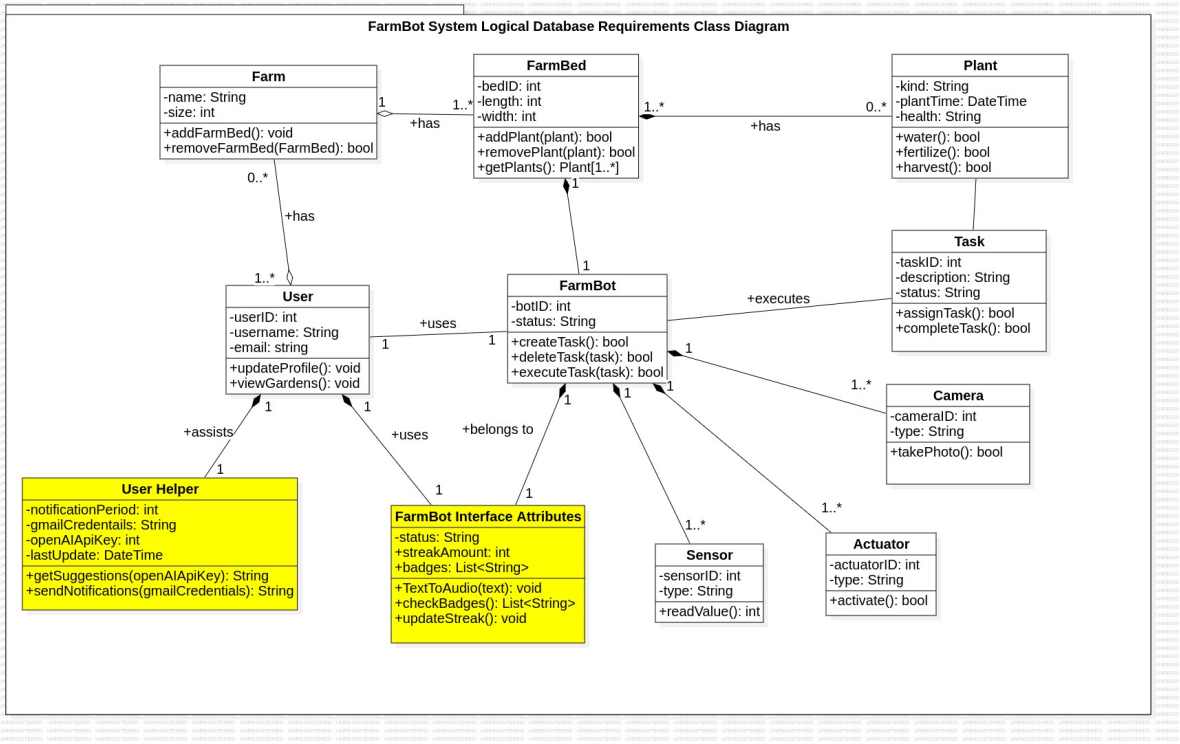

Figure 4.7: Logical Database Diagram (Suggested)

4.5 Design Constraints

- Since we use external sources such as OpenAI and Gmail, we must ensure that user's personal data should be kept securely in those systems with encryption and authorization.
- With additional features, FarmBot system relies on external services provided by OpenAI and Google. We must ensure that system will handle the possible external service downtimes properly.
- Any information stored in FarmBot system shall comply with the regulatory policies such as Personal Data Protection Law (KVKK).

4.6 System Quality Attributes

4.6.1 Usability

- System shall be available in each platform to let users use FarmBot with the help of Gmail API.
- Users shall easily manage FarmBot with the help of OpenAI, Gmail API and FarmBot WebApp.

4.6.2 Performance

- Gmail API ensures on time delivery of farm summary updates, minimizing delays and ensuring that users receive relevant information.
- Optimizing the integration with the OpenAI API to ensure quick response times for queries helps improved performance, enhancing the overall user experience.

4.6.3 Reliability

- The stability and reliability of interactions with the OpenAI and Gmail APIs is essential to maintain the overall reliability of the FarmBot system, minimizing the risk of service disruptions or errors.
- FarmBot system must implement robust error handling mechanisms to ensure reliable in the event of netweork issues or API failures.

4.6.4 Availability

- Monitoring the uptime and availability of the OpenAI and Gmail APIs ensures continuous accessibility to key functionalities within the FarmBot system, minimizing downtime and interruptions to user workflows.

4.6.5 Security

- Implementing secure authentication mechanisms for accessing the OpenAI and Gmail APIs helps protect sensitive user data and prevent unauthorized access to system functionalities.

4.7 Supporting Information

Although the FarmBot system is able to work by itself, the periodic check on hardware should be done to ensure proper working.