Department of Computer Engineering

CENG350 Software Engineering

Software Requirements Specification for FarmBot

Group 56

By

Hasan Küreli

Yusuf Sami Lök

Saturday 6th April, 2024

Contents

List of Figures

List of Tables

Revision History

Date	Description	Version
24.03.2024	Initializiation	0.0.1
29.03.2024	SRS Part-1	1.0.0
06.04.2024	SRS Final Part	1.0.0

Table 1: Revision History of Software Requirements Specification Document

1. Introduction

1.1 Purpose of the System

FarmBot is a flexible and effective instrument for precision farming that is intended to automate and optimize agricultural activities. FarmBot seeks to reduce labor and environmental impact while streamlining crop cultivation, optimizing resource consumption, and improving yields through the integration of cutting-edge technology with conventional agricultural methods. It makes it possible to sow, irrigate, weed, and harvest crops precisely, guaranteeing that each plant receives the right amount of space, time, and resources. Farmers can adopt flexible and adaptive farming practices by tailoring FarmBot's operations to their own crop varieties, planting times, and environmental conditions. FarmBot boosts farming efficiency and lowers the need for manual labor by automating repetitive chores and scheduling operations. This allows users to manage a greater area of land with less resources.

Reduced water consumption, less chemical inputs, and targeted treatments by FarmBot maximize plant health and promote resource stewardship and environmental conservation. Apart from its practical applications, FarmBot serves as a teaching platform covering robotics, technology, and agriculture. It encourages creativity by letting users try out cutting-edge farming techniques, sensor technologies, and automation schemes.

In the end, FarmBot hopes to revolutionize agriculture by empowering indi-

viduals and groups to cultivate food in a productive, sustainable, and independent way. Food security, environmental sustainability, and technical innovation in farming practices would all benefit from this.

1.2 Scope

Software products work together to enable users to automate and manage various aspects of their farming operations using FarmBot hardware. Here are:

- **FarmBot OS**: With the help of this software, which acts as the FarmBot hardware's operating system, automated farming operations may be controlled and managed.
- **FarmBot Web Interface**: Through this web application, users can schedule tasks, schedule interactions with the FarmBot system, schedule tasks, check progress, and get notifications.
- **FarmBot Mobile Application**: A mobile application improves the online interface and gives users greater flexibility to operate and keep an eye on FarmBot from their smartphones or tablets.

Using an easy-to-use interface, you can manage FarmBot's motions and functions, such as planting, watering, weeding, and harvesting. Adjust planting dates, watering schedules, and upkeep assignments according to crop varieties, local conditions, and user preferences. Get up-to-date information, notifications, and alerts in real-time on tasks completed, system health, and possible problems that need to be fixed. To evaluate progress, assess trends, and make well-informed decisions for optimizing farming methods, access sensor readings, historical data, and farming analytics.

The application of the specified software aims to simplify agricultural procedures by boosting farming productivity, eliminating the need for manual labor, and automating repetitive operations. Encourage precision farming by increasing agricultural yields,

reducing waste, and making the most use of available resources. Promote soil health through targeted treatments, minimize water use, and decrease chemical inputs to foster sustainability and environmental stewardship. Give people and communities the tools they need to grow food in a sustainable, self-sufficient, and efficient manner, thereby promoting environmental preservation and food security.

This scope aligns with higher-level requirements, such as system requirements, which outline the functionality, performance, and interoperability requirements for the FarmBot system. It also fits in with the larger goals and objectives of the FarmBot project, which prioritize innovation, education, and empowerment in the realms of agriculture and technology.

The business domain under consideration is agricultural technology, specifically focusing on precision farming solutions.

The range of business activities included in this business domain encompasses various divisions and functions related to modern agricultural practices:

- **Crop Cultivation**: Activities related to planting, watering, weeding, and harvesting crops.
- **Resource Management**: Management of resources such as water, soil nutrients, and pesticides to optimize crop growth and yield.
- **Equipment Management**: Maintenance and operation of farming equipment, including tractors, irrigation systems, and precision farming tools.
- **Data Analysis and Decision Making**: Analysis of sensor data, weather patterns, and crop performance to make informed decisions regarding planting, irrigation, and pest control.
- **Environmental Monitoring**: Monitoring of environmental factors such as soil moisture, temperature, and humidity to assess crop health and identify areas for improvement.

The FarmBot system, an automated precision farming solution, is the scope of the system being built or modified within this business domain. Crop cultivation, resource management, equipment management, data analysis, and decision-making are all supported by the system. It is presumed that the system will mainly concentrate on automating chores like planting, watering, and weeding in addition to offering real-time control and monitoring features to maximize farming operations. To improve functionality and aid in decision-making, the system may also link with outside organizations like weather services, soil testing facilities, and agricultural marketplaces.

1.3 System Overview

1.3.1 System Perspective

FarmBot is a stand-alone system, so it is not a part of a larger system. It is an autonomous farming system, seamlessly integrates with cloud-based agricultural platforms like AgriCloud Services for weather forecasting and soil analysis. Its user interfaces cater to diverse stakeholders, from farm operators monitoring activities through the mobile app to field managers accessing analytics via a web dashboard. Through these interactions, FarmBot revolutionizes farming by enhancing decision-making, optimizing resource usage, and driving sustainability.

1.3.1.1 System Interfaces

To connect and communicate with each other, the hardware, software, and users of the FarmBot system require a range of interfaces. Through the motor interface, the tool control software is linked to the motors that propel movement in the x, y, and z axes. It transmits data such as sensor input, tool engagement, and movement orders. A variety of sensors, including temperature and soil moisture sensors, are connected to the program through the Sensor Interface, which also permits data transfer and calibration

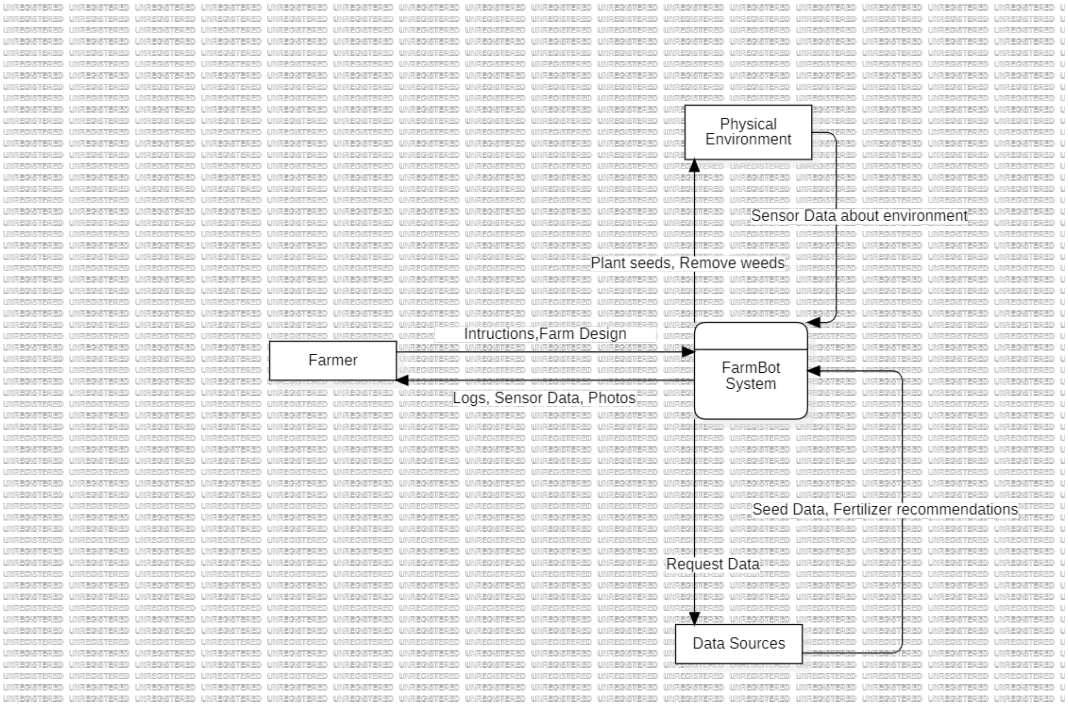

Figure 1.1: Context Diagram

instructions between the sensors.

To effectively manage power distribution, the Power Interface transmits information regarding voltage levels and power conditions. The connection interface manages commands, status updates, and data synchronization in addition to facilitating communication with external devices and networks. Users can utilize a graphical or command-line interface to plan jobs, change configurations, and track their progress through the system's User Interface (UI). Through the Control Software Interface, it communicates commands, sensor data, and status updates.

The Cloud Service Interface facilitates the sharing of sensor data, system logs, synchronization instructions, analytics, remote monitoring, software updates, and data storage when paired with cloud-based services. Not to add, you may access maintenance and support services for updates, repairs, and troubleshooting through the Maintenance and Support Interface, which also allows you to share error logs, diagnostic data, and support requests. These interfaces facilitate user communication, system

maintenance, and effective operation within the context of the FarmBot ecosystem.

1.3.1.2 User Interfaces

An intuitive graphical user interface facilitates communication between users and the agriculture management software. In addition to monitoring historical and real-time sensor data (temperature, soil moisture), accessing reports for well-informed decision-making, and interacting with visualizations to comprehend crop health and resource utilization, farmers and specialists can customize Farmbot settings (planting timetables, watering routines). The interface places a high priority on responsiveness across devices, accessibility (meeting a range of demands), and ease of use. By specifying visual elements, interaction design, and content organization for a consistent and understandable user experience, a dedicated style guide can further hone these qualities.

The web interface of FarmBot is truly user-friendly. In this application, the first thing we encounter is an area where we will grow our plants. This area may vary in size depending on the model of the product we use. The area is divided according to the x, y, and z axes, allowing us to drag and drop our seeds wherever we want to plant them. The x and y axes are arranged as if looking down on the field, while the z axis indicates the height. Fiqure 1.2 is an introduction to the web application:

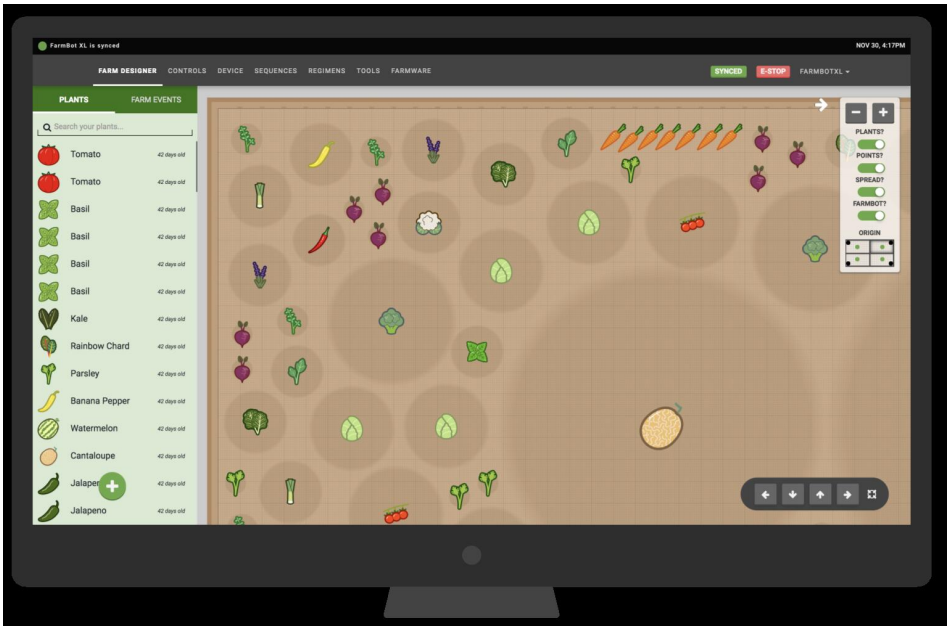

Figure 1.2: Quick Introduction to User Interfaces

The toolbar menu, accessed from the triple bar icon in the upper left corner, greatly facilitates users' tasks. Here, you can view the map or adjust the colors of plant varieties. You can also select the age of the plants to be displayed in the simulator. In addition to these, users can adjust sensors, graphs, the status of machines, watering intervals, and many other features according to their preferences. Figure 1.3 illustrates the menu.

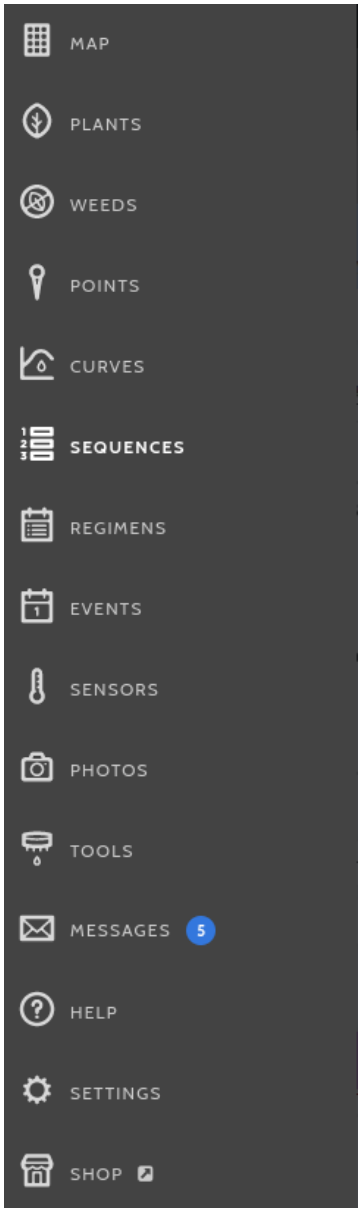

Figure 1.3: Toolbar Menu

As seen in Figure 1.4 and Figure 1.5, there is a broccoli in the middle, and a spinach. There is data available regarding how far away from each plant you can plant another plant. As shown here, the distance range for spinach is lower compared to broccoli. As we bring spinach closer to broccoli, the color starts to change from green to red. The circles around the plants indicate the area that defines the plant's space.

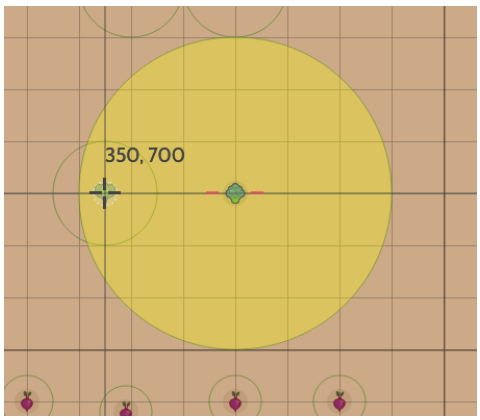

Figure 1.4: Spinach is away from broccoli

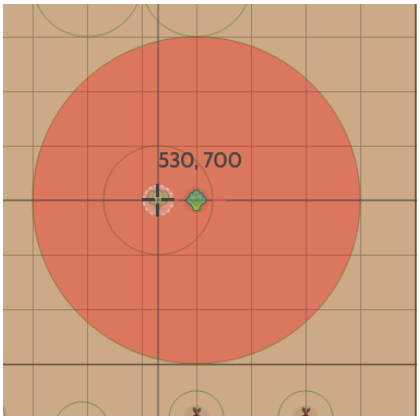

Figure 1.5: Spinach is close to broccoli

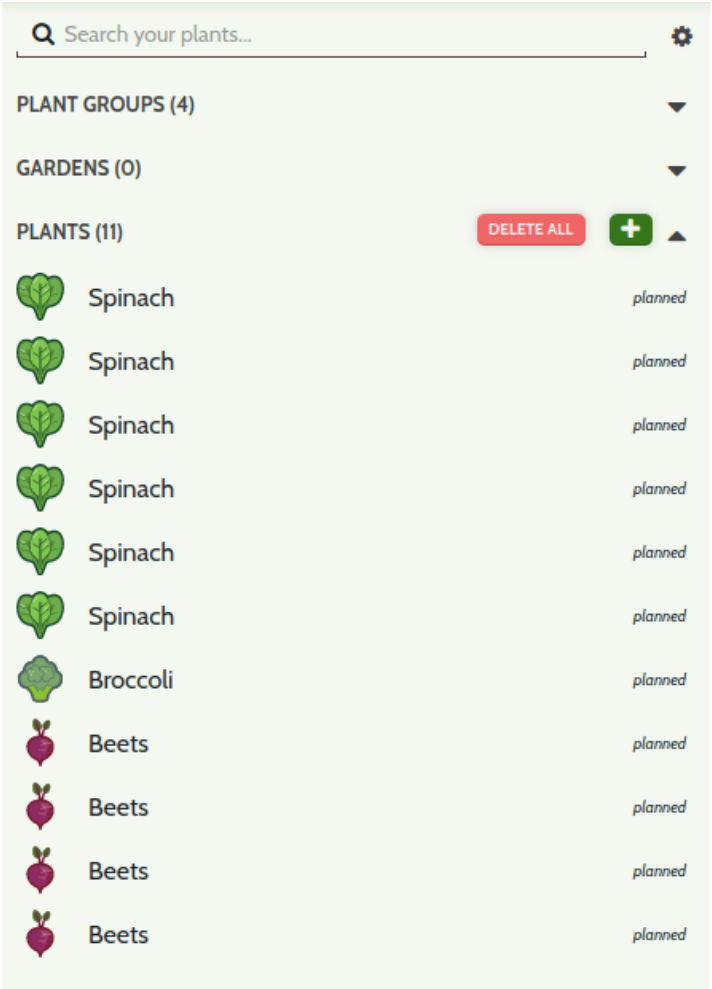

Figure 1.6: Plant Section

In the plant section, the desired type of plant can be added to the inventory for planting in the desired amount of space. This interface has been designed in a very easy and understandable manner. Previously designed plant groups can also be used, eliminating the need to select them repeatedly. An example of the plant section can be found in Figure 1.6. Additionally, there is a button available for deleting all plants.

1.3.1.3 Hardware Interfaces

In this section, we will specify the logical characteristics of each interface between the software product and the hardware. A hardware interface determines how hardware is controlled according to a specific standard or protocol.

FarmBot Genesis has the capability to autonomously switch tools to execute different tasks thanks to the Universal Tool Mount, or UTM. FarmBot's Universal Tool Mount, which has multiple logical features to enable smooth integration and operation, is a critical interface between the hardware and software components of the system. Its flexible architecture allows it to handle a wide range of tools, each requiring a unique set of port configurations and instruction sets for attachment and use. In order to communicate with supported instruments efficiently, the mount uses common communication protocols like UART and I2C. It supports a variety of agricultural devices, including soil sensors and seed injectors.

Figure 1.7: Universal Tool Mount

By interacting with the FarmBot software platform, it translates commands into concrete actions and dynamically modifies its configuration to accommodate different activities. Additionally, it provides the program with real-time input regarding its condition and errors discovered, ensuring precise and fruitful agricultural operations. Essentially, the Universal Tool Mount is critical to FarmBot's mission of revolutionizing modern agriculture through intelligent automation and sustainable practices.

Another essential component of Farmbot technology is the seeder. It works by using a vacuum pump to suction-hold a single seed at the end of a luer lock needle. Different sized needles can be used for different sized seeds to improve sowing efficiency and consistency.

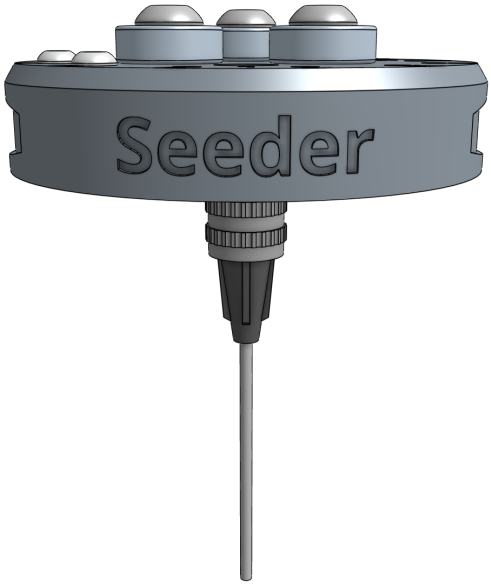

Figure 1.8: Seeder Tool

A key component of FarmBot, the Seeder was created to automate the precise and effective planting of seeds. With its versatile and adaptable design, the Seeder smoothly connects with FarmBot's framework to enable users to automate the process of seeding a wide range of crops. With the use of cutting-edge technology, the Seeder can precisely dispense seeds at pre-established depths and intervals, guaranteeing ideal spacing and germination rates. By reducing seed loss and increasing crop output, this novel tool not only saves time and effort but also encourages sustainable agricultural practices. By enabling farmers and gardeners to develop healthy and bountiful harvests with minimal work and offering an intuitive interface and customizable settings, the Seeder represents a substantial improvement in agricultural production and automation.

Another important tool is Watering Nozzle. It accepts a concentrated stream of water coming from the UTM and turns it into a gentle shower for the plants.

Figure 1.9: Watering Nozzle

Accurate and efficient crop irrigation is made possible by the Watering Nozzle, an essential part of the FarmBot system. Its well-considered logical characteristics ensure optimal usability and performance by facilitating a seamless hardware-software interface. Due to its flexible design, the Watering Nozzle could require specific port configurations to connect to the FarmBot hardware and enable easy installation and usage. The FarmBot software platform provides instructions that control things like irrigation target areas, duration, and water flow rate. It responds to these commands. It is compatible with a particular set of instruction sets. The Watering Nozzle is designed to work with a variety of irrigation equipment, such as sprayers, misters, and drip emitters, each of which has its own control and communication protocols. The Watering Nozzle efficiently communicates with the software via common protocols like UART or special protocols made to fit FarmBot's needs. This allows it to give real-time feedback on irrigation status and any problems that may have occurred. Within the FarmBot ecosystem, this logical integration guarantees accurate and effective irrigation, which improves crop health and yield.

1.3.1.4 Software Interfaces

FarmBot requires several key software components and interfaces to function effectively. At its core is FarmBot OS (FBOS), the operating system that manages the entire FarmBot system.

FarmBot OS: Using the message broker, FarmBot's Raspberry Pi operates a customized operating system called FarmBot OS, which keeps the connection and synchronization with the web application. This enables real-time control, the downloading and execution of planned actions, and the uploading of logs and sensor data by FarmBot. The OS sends G and F code commands to the Farmduino/Arduino via a USB cable or serial connection, and it also receives data that has been gathered from sensors and rotary encoders.

Using a built-in tool called configurator, FarmBot OS makes it simple to enter web app and WiFi credentials from a WiFi-enabled device (such a laptop or smartphone). This is what you use to set up your FarmBot the first time, to link it to your home WiFi and online app account.

Arduino Firmware: The firmware that is installed on the Arduino or Farmduino microcontroller physically controls the motors, tools, sensors, and other electronics of FarmBot. In line with the G and F codes that it receives from FarmBot OS, it rotates the motors and reads and writes pins. It also transmits the data collected from the rotary encoders and pin readings to the Raspberry Pi.

The Farmduino Firmware, which functions as the real-time component managing hardware interactions, is a complement to FBOS. FarmBot uses PostgreSQL, a powerful database management system, to manage data.

1.3.1.5 Communication Interfaces

For efficient control and data transfer, FarmBot uses a range of communication interfaces for local network protocols. These include Ethernet and Wi-Fi connections to enable wired and wireless communication with other devices on the local network.

The suite of TCP/IP protocols ensures consistent and reliable data delivery while facilitating communication. Moreover, FarmBot makes use of HTTP to streamline communication between the FarmBot OS and the web interface, providing users with a seamless browsing experience. For efficient machine-to-machine communication, MQTT offers lightweight messaging, while WebSocket allows real-time updates and control. Together, these interfaces enable communication between humans, sensors, actuators, and other local network systems, hence enhancing FarmBot's utility and integration potential in many farming contexts.

1.3.1.6 Memory Constraints

The characteristics and limitations of primary and secondary memory impact FarmBot's functionality. Program instructions, temporary data, and running processes must all be stored in primary memory. Typically, RAM (Random Access Memory) is used for it. This memory allows the FarmBot OS and apps to get data fast, guaranteeing responsiveness and smooth operation during farming. However, the exact hardware configuration may affect the available RAM, and capacity issues may make it more difficult for the system to efficiently handle multiple tasks running at once.

Similar to primary memory, secondary memory—like flash memory or onboard storage—is used to store configuration files, application software, user-generated data, and critical system parts. Secondary memory serves as permanent storage for FarmBot OS; however, hardware restricts the amount of data that can be saved locally on the device. Large-scale dataset operations like firmware updates or logging may have an impact on system performance due to read/write rates in secondary memory that affect data

access and retrieval times. To maximize FarmBot's performance and guarantee reliable operation in a variety of agricultural scenarios, it is necessary to comprehend these features and constraints.

1.3.1.7 Operations

Users can initiate actions like watering, weeding, planting, and harvesting using the Users can initiate actions like watering, weeding, planting, and harvesting using the FarmBot web interface or mobile application. These operations entail direct user input and control over FarmBot's behavior.Users can plan automated tasks based on the time of day, the weather, or special occasions. This entails setting up the FarmBot to perform maintenance tasks, watering schedules, and planting schedules on its own.Users do diagnostic and maintenance tasks to ensure FarmBot is functioning as intended. This can mean fixing hardware issues, calibrating sensors, and updating firmware.

Using the web interface or mobile application, users may communicate with FarmBot in real-time to track developments, make adjustments, and start human operations when necessary. This includes viewing real-time video feeds, managing FarmBot's actions, and keeping an eye on sensor data. FarmBot runs on its own and needs little assistance from the operator to finish its assigned chores. Users may receive notifications or messages about task completion, system status, or any issues that need to be addressed while the operation is being performed without their presence.

FarmBot logs data, such as sensor readings, task completion times, and system events, for analysis and reporting. Users can access historical data to track farming progress, identify trends, and make informed decisions. FarmBot helps customers better understand and manage their agricultural projects by providing graphical representations of sensor data, task schedules, and farm layouts.

In the case of a system failure or hardware malfunction, users can create backups of important data, such as configuration settings, planting schedules, and history logs, to avoid data loss. Users can use backups to restore FarmBot to a previous state in case of a system problem or data damage. To restore normal operation, this can include importing the stored data, changing the settings, and reinstalling the firmware.

1.3.2 System Functions

Table 1.1: Major Functions of the Software

Function	**Description**
Planting	Enable users to plan and schedule planting of crops.
Watering	Set up watering schedules and control irrigation systems to ensure proper hydration of crops.
Weeding	Facilitate automated weeding processes to remove unwanted plants and maintain crop health.
Harvesting	Schedule and manage harvesting activities to collect mature crops.
Monitoring	Continuously monitor environmental factors such as soil moisture, temperature, and humidity to assess crop conditions.
Control	Remotely control FarmBot operations, including movement and tool engagement, through a web interface or mobile application.
Data Analysis	Analyze sensor data and historical records to provide insights into crop performance and identify areas for improvement.
Reporting	Generate reports summarizing farming activities, crop yields, and environmental conditions for analysis and decision making.

1.3.3 Stakeholder Characteristics

The intended users of the FarmBot system encompass a diverse group of individuals involved in agriculture, including farmers, agricultural technicians, and enthusiasts. Their characteristics vary widely, including educational level, experience in farming practices, technical expertise, and potential disabilities. Understanding these general characteristics is crucial for designing a user-friendly system that accommodates the needs of all users.

1. **Educational Level:** Users may have varying educational backgrounds, ranging from minimal literacy to advanced degrees in agriculture or technology. This has an impact on their comprehension and proficiency with the system interface and documentation.

2. **Experience:** Users may range in experience from beginners to seasoned pros when it comes to farming techniques. Their ability to operate the FarmBot system efficiently is influenced by their knowledge of agricultural technology and procedures.

3. **Disabilities:** Certain users may be affected by disabilities that affect how they engage with the system, such as cognitive, mobility, or vision impairments. To ensure that the system is inclusive of all users, it is imperative to incorporate accessibility features and usability concessions.

4. **Technical Expertise:** Users may have varying degrees of technical proficiency, from simple computer literacy to sophisticated programming abilities. This affects both their comfort level with using digital interfaces and their capacity to configure and troubleshoot the system.

Stakeholders involved in the development and operation of the FarmBot system:

1. **Farmers:** The system's main users who gain from improved farming methods' sustainability, productivity, and efficiency.

2. **Agricultural Organizations:** Organizations that assist agricultural innovation, research, and extension services may be able to offer resources and experience to aid in the creation and uptake of new systems.

3. **Software Developers:** The persons or groups in charge of creating, executing, and preserving the FarmBot software, guaranteeing its efficiency, dependability, and safety.

4. **Regulatory Authorities:** Governmental or private organizations in charge of establishing guidelines, rules, and policies pertaining to data privacy and agricultural technology, which may have an effect on the deployment and design of systems.

1.3.4 Limitations

a) Regulatory requirements and policies: The supplier's options may be limited by regulations governing agricultural technology, data privacy, and environmental protection, which must be adhered to during system development and operation.

b) Hardware limitations: The system must operate within the constraints of the FarmBot hardware, including signal timing requirements and compatibility with sensors, actuators, and other components.

c) Interfaces to other applications: Compatibility with existing agricultural software systems and protocols must be considered to ensure seamless integration and interoperability.

d) Parallel operation: The system should support parallel operation of multiple FarmBot units, enabling simultaneous farming activities across different areas of land.

e) Audit functions: The system may require built-in audit functions to track user actions, system events, and data modifications for compliance and accountability purposes.

f) Control functions: The system must provide robust control functions to manage FarmBot operations, including movement, tool engagement, and task scheduling, while ensuring safety and efficiency.

g) Higher-order language requirements: The system may need to support programming languages or scripting interfaces for advanced customization and automation of farming tasks.

h) Signal handshake protocols: Proper signal handshake protocols, such as XON-XOFF or ACK-NACK, may be necessary for reliable communication between FarmBot and external devices or systems.

i) Quality requirements: The system must meet specific quality standards, including reliability, accuracy, and durability, to ensure consistent performance and user satisfaction.

j) Criticality of the application: The system's reliability and robustness are crucial, as any failure or malfunction could have significant consequences for crop cultivation and farm productivity.

k) Safety and security considerations: Measures must be implemented to address safety risks, such as collision avoidance, equipment malfunction, and unauthorized access, as well as protect sensitive data from cybersecurity threats.

l) Physical/mental considerations: The system's user interface should be designed with ergonomic principles in mind to accommodate users with physical disabilities or limitations, ensuring ease of use and accessibility.

m) Limitations sourced from other systems: Real-time requirements imposed by controlled systems, such as irrigation systems or environmental sensors, must be considered when designing interfaces and coordinating system operations to ensure timely and accurate response to changing conditions.

1.4 Definitions

Term	Definition
UI	User Interface, the interface through which users interact with a software system.
API	Application Programming Interface, a set of protocols, tools, and definitions that allows different software applications to communicate with each other.
OS	Operating System, software that manages computer hardware and provides services for computer programs.
IoT	Internet of Things, a network of interconnected devices that can communicate and exchange data over the internet.
HTTP	Hypertext Transfer Protocol, the protocol used for transmitting data on the World Wide Web.
HTTPS	Hypertext Transfer Protocol Secure, an extension of HTTP that provides secure communication over a computer network.
Seeder	A tool attached to FarmBot for planting seeds in the soil with precision and accuracy.
Watering Nozzle	A device used by FarmBot to deliver water to plants at specific locations and in controlled amounts.
Universal Tool Mount	An element of FarmBot that allows for the attachment and interchangeability of various tools and implements, such as the seeder and watering nozzle.
MQTT REST API	Message Queuing Telemetry Transport Interface for Application Programming with Representational State Transfer

[1]

2. References

[1] FarmBot Inc., "FarmBot - Open-Source CNC Farming," 2022. https://farm.bot/.

[2] IEEE, "ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering," *ISO/IEC/IEEE 29148:2018(E)*, Nov. 2018.

3. Specific Requirements

3.1 External Interfaces

Interfaces for cloud integration, control software communication, user engagement, and maintenance assistance are all included in the FarmBot system. Users can monitor status and control the system with the help of the User Interface (UI). The UI and control software can communicate more easily thanks to the Control Software Interface. For data interchange and updates, the system is integrated with cloud-based services via the Cloud Service Interface. Access to assistance and troubleshooting services is made possible through the Maintenance Support Interface. When combined, these interfaces guarantee the FarmBot system operates and is managed with ease.

User Interface (UI)

The FarmBot system encompasses several interfaces to ensure its seamless operation and management. The User Interface (UI) serves as the primary point of interaction for users, allowing them to control the system, monitor status, and adjust settings. The Control Software Interface facilitates communication between the UI and the underlying control software, enabling commands, sensor data, and system status updates to be exchanged. The Cloud Service Interface integrates the FarmBot system with cloud-based services for data exchange, remote monitoring, and software updates, enhancing scalability and functionality. The Maintenance Support Interface provides access to maintenance and support services, allowing users to troubleshoot issues, send diagnostic data, and receive assistance for efficient system operation and maintenance. Together,

these interfaces form a comprehensive ecosystem to streamline farming automation and ensure optimal performance of the FarmBot system.

The main point of contact between people and the automated farming system is the FarmBot system's User Interface (UI). It includes all of the graphical components and user interface elements, like input fields, sliders, and buttons. The user interface (UI) is in charge of providing users with pertinent data, such as sensor readings, system alarms, and the status of ongoing operations. The user interface (UI) allows users to enter commands, adjust settings, and track the advancement of farming operations. The UI improves the entire user experience and makes it possible for users to efficiently manage and control the FarmBot system by offering an intuitive and user-friendly interface.

Control Software Interface

Between the FarmBot system's User Interface and its underlying control software, the Control Software Interface serves as a link. It makes it easier for the control software and user interface to communicate commands, sensor data, and system status updates. Through this interface, users can receive real-time sensor data for monitoring and analysis, submit commands from the user interface (UI) to the control software for execution, and keep track of the system's present condition. Users may engage with the FarmBot system effectively and efficiently thanks to the Control Software Interface, which maintains seamless connection between the UI and the control software.

Cloud Service Interface

Access to maintenance and support services for software updates, repairs, and troubleshooting is made possible through the Maintenance Support Interface. It lets customers to send diagnostic data for analysis, seek for help from maintenance support staff, and get advice on fixing problems or doing maintenance. This interface makes it easier for users and support personnel to communicate effectively, which expedites

the process of resolving technical problems and guarantees the uninterrupted operation of the FarmBot system. The automated farming system's lifetime, availability, and dependability are all improved by the Maintenance Support Interface's extensive maintenance and support capabilities.

Maintenance Support Interface

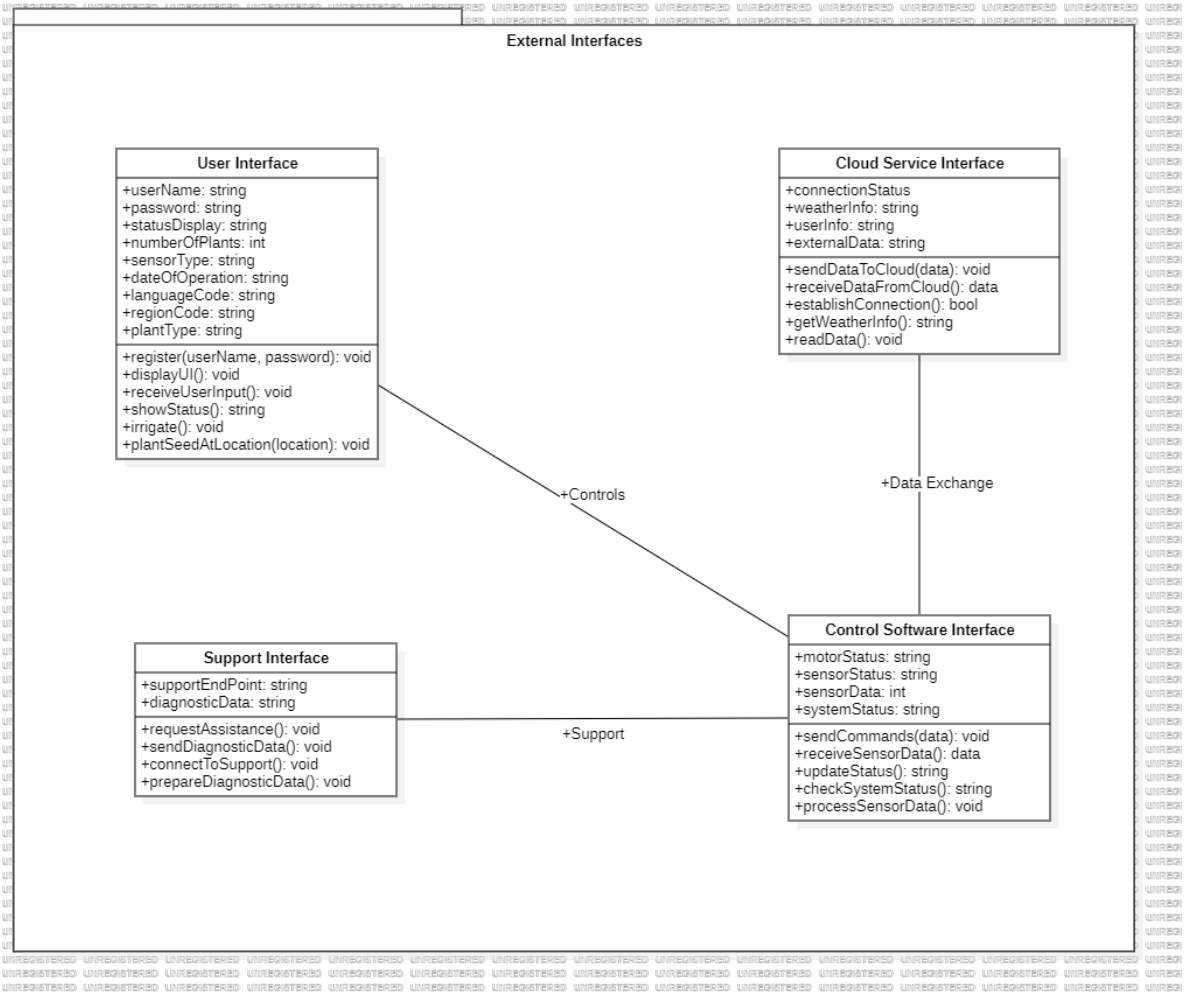

Figure 3.1: External Interfaces Class Diagram

3.2 Functions

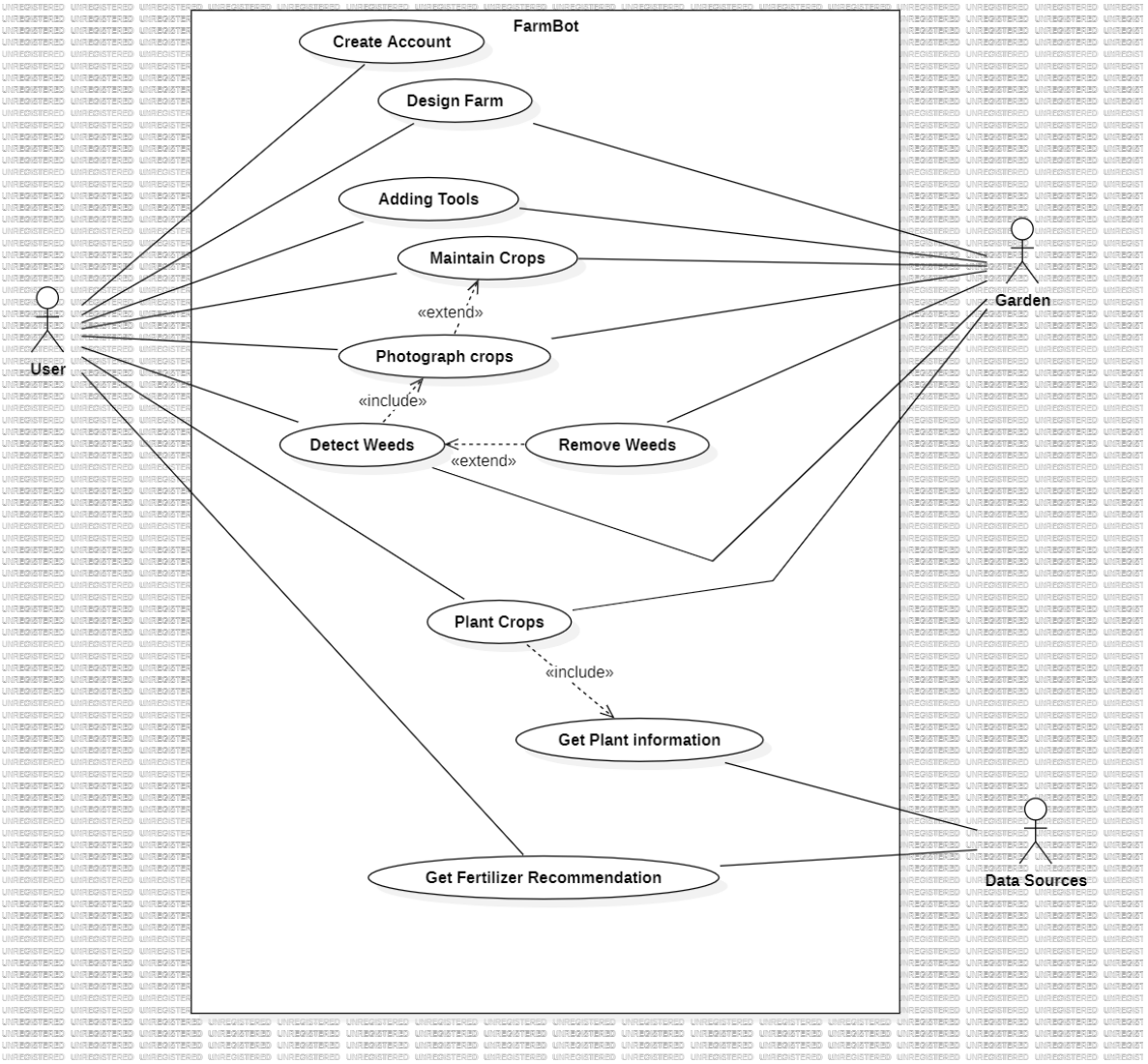

Figure 3.2: Use Case Diagram

Use Case	Create Account
Actors	User
Description	User creates a new account for the FarmBot system through the app. This grants access to FarmBot functionalities and data storage.
Preconditions	User has access to the FarmBot app and an internet connection.
Data	User-provided information (email, password, name).
Stimulus	User initiates account creation within the app.
Normal Flow	1. User opens the FarmBot app and selects "Create Account". 2. User enters required information (email, password). 3. User agrees to terms of service (if applicable). 4. User submits the account creation request. 5. The app validates the information and creates the account in the FarmBot system. 6. The app provides confirmation and gives a quick guide to the user.
Exception Flow	Communication errors between the app and FarmBot server. Invalid or missing user-provided information where user must re enter valid information.
Post Conditions	User has a new FarmBot account and can access the system's functionalities.

Table 3.1: Create Account Use Case for FarmBot

Use Case Name	Design Farm
Actors	User, Gardem
Description	The farmer defines the layout of their farm plot for FarmBot Express to operate in. This includes specifying plot dimensions (length, width), defining planting zones, selecting crops from the app, setting basic planting parameters, saving the design for future reference within the FarmBot app.
Preconditions	FarmBot Express system is set up and operational.
Data	Farm plot dimensions (length, width), crop selection data, planting parameter data (row spacing, planting depth)
Stimulus	Farmer interacts with the FarmBot Express app to define the farm plot design.
Normal Flow	1. Farmer opens the FarmBot Web App. 2. Farmer specifies his/her FarmBot Model (Express for our case). 3. Farmer defines planting zones. 4. Farmer selects crops they want to add to their design from the database. 5.Farmer sets basic planting parameters (row spacing, planting depth). 6. Farmer saves the design for future reference. 7. FarmBot Express app stores the design data.
Alternative Flow	-
Exception Flow	* If invalid data is entered (e.g., negative plot dimensions), the app prompts the farmer to re-enter valid values. * If communication errors occur (app to system), the farmer might receive a notification.
Post Conditions	The farm plot design is saved in the FarmBot Express app and can be referenced for future use.

Table 3.2: Design Farm Use Case for FarmBot Express

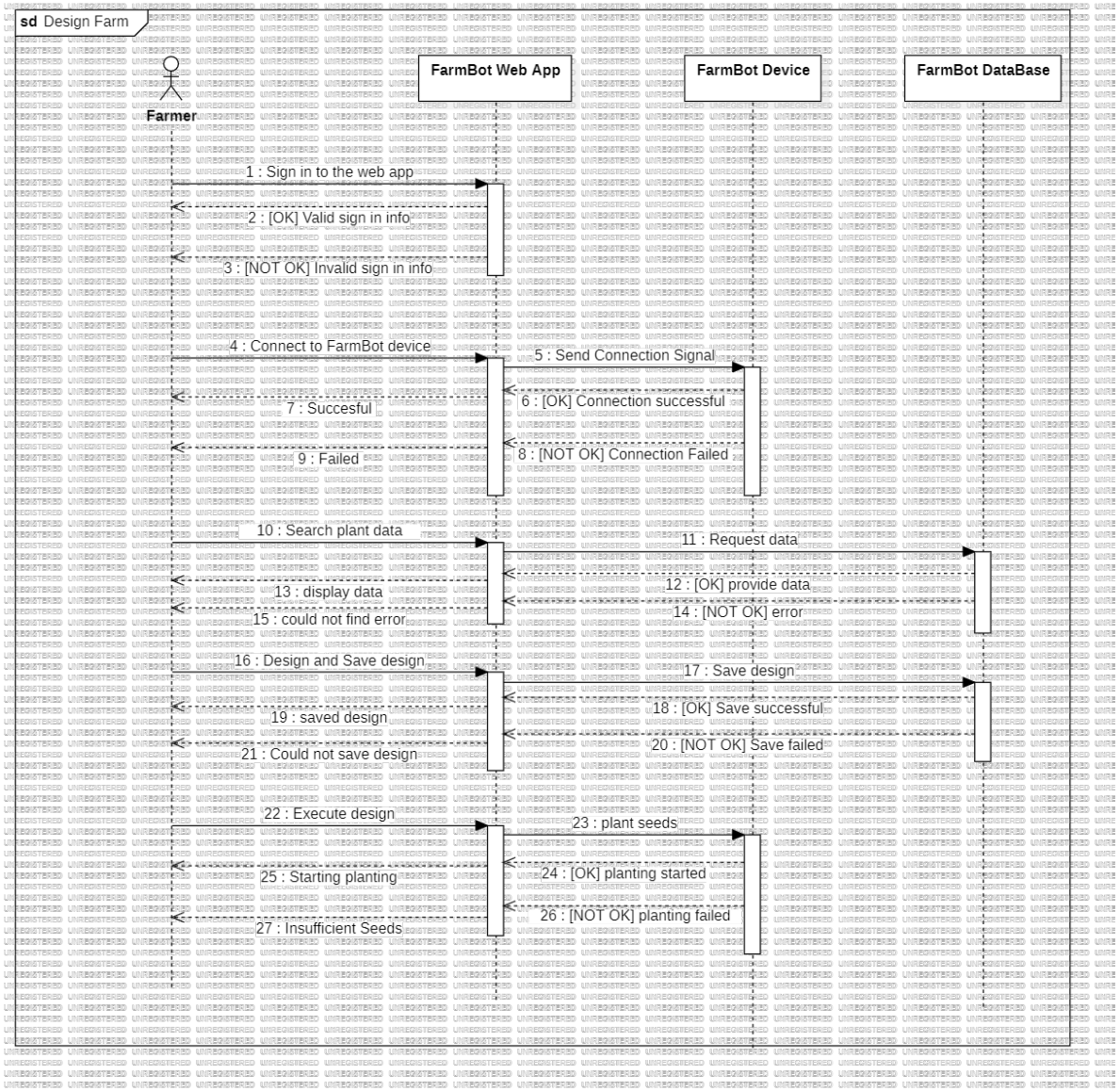

Figure 3.3: Sequence Diagram for Design Farm Use Case

Use Case Name	Photograph Crops
Actors	User, Device (FarmBot), Gardem
Description	The FarmBot device takes pictures of the user's crops when the user starts the process. This can be used to track the development of crops, identify weeds, and monitor crop health.
Preconditions	FarmBot device is operational, powered on and has a functional camera.
Data	User-defined settings, number of photographs to capture.
Stimulus	User interacts with the FarmBot control interface (web app, mobile app) to start the photograph crops process.
Normal Flow	1. User selects the "Photograph Crops" function within the FarmBot app. 2. User defines settings for the photograph capture. 3. User initiates the process by sending a command to the FarmBot device. 4. FarmBot device receives the command and prepares for image capture. 5. FarmBot device moves itself to specified locations within the farm plot and captures images. 6. Captured photographs are stored on the FarmBot device.
Alternative Flow	6. Captured photographs are transferred to a designated storage location (e.g., user's cloud storage).
Exception Flow	If the FarmBot device encounters obstacles during movement (e.g., plants blocking path), the user might be notified and corrective action might be required.
Post Conditions	Captured photographs are stored on the FarmBot device or a designated location and the user has access to the photographs for further analysis or record-keeping.

Table 3.3: Photograph Crops Use Case for FarmBot

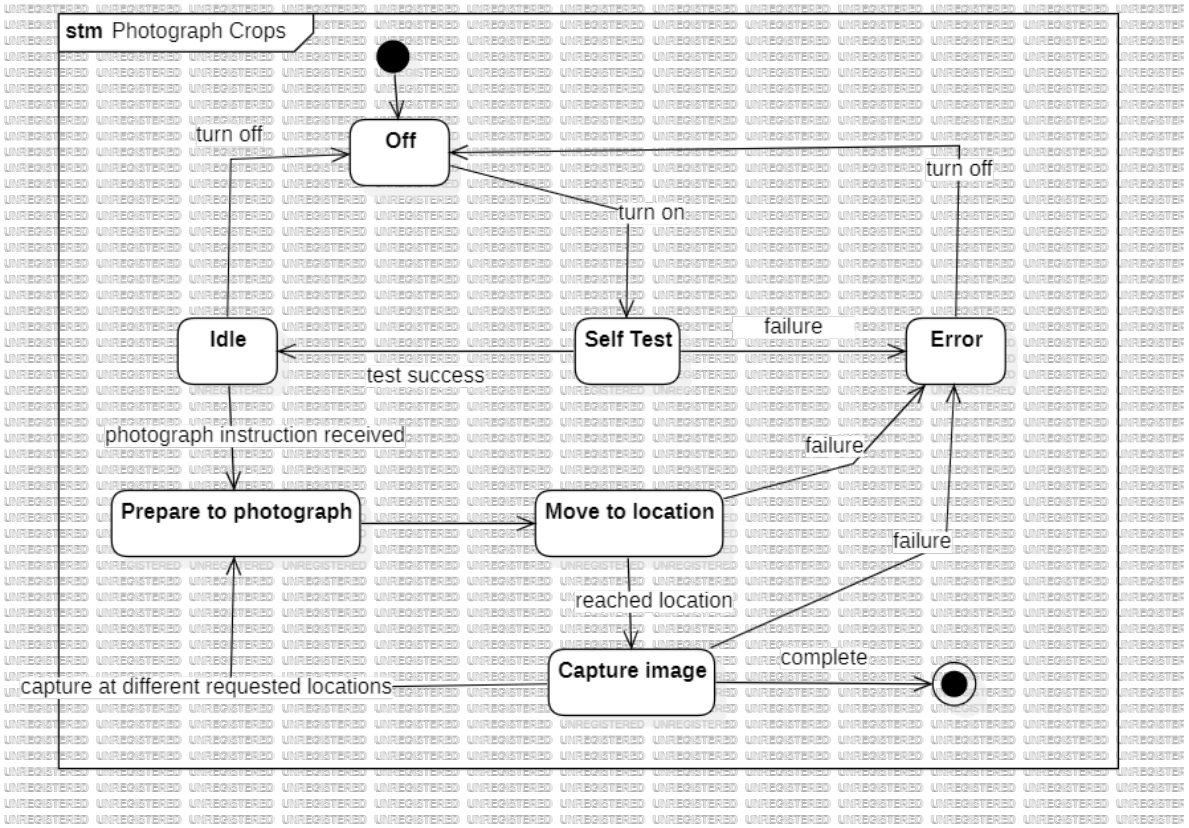

Figure 3.4: State Diagram for Photograph Crops Use Case

Use Case Name	Maintain Crops
Actors	User, Gardem
Description	Plan & schedule tasks (watering, weeding, fertilizing) and monitor plant health through the FarmBot App. FarmBot can automate tasks and send alerts.
Preconditions	FarmBot device is operational, sensors functional.
Data	User-defined tasks, sensor data (optional).
Stimulus	User interacts with FarmBot App.
Normal Flow	1. User plans tasks in FarmBot App. 2. Defines settings for automated tasks. 3. App stores schedule. 4. FarmBot executes tasks (automated) or waits for user intervention. 5. User monitors plant health (photos/sensor data). 6. App sends notifications. 7. User tracks history.
Alternative Flow	User notified of abnormal sensor data (e.g., low battery).
Exception Flow	Communication errors or task completion issues trigger user notification.
Post Conditions	Tasks completed, plant health data available, history logged.

Table 3.4: Maintain Crops Use Case for FarmBot

Use Case Name	Add Tools
Actors	User, Gardem
Description	This use case covers the process of adding new tools to the FarmBot system through the FarmBot app. These tools can be physical attachments or software integrations that expand FarmBot's functionalities.
Preconditions	FarmBot device is operational and connected to the internet.
Data	User selection of the tool to be added (from available options).
Stimulus	User interacts with the FarmBot app to add a new tool.
Normal Flow	1. User opens the FarmBot app and navigates to the tool management section. 2. The app displays a list of available tools compatible with the user's FarmBot device. 3. User selects the desired tool to add. 4. The app provides instructions for attaching the tool to the FarmBot device. 5. The FarmBot app updates its internal configuration to recognize the newly added tool.
Alternative Flow	4. The app might guide the user through an installation process or request access codes to integrate the tool.
Exception Flow	If the selected tool is unavailable or incompatible with the user's FarmBot model, the app might display an error message or recommend alternative options.
Post Conditions	The new tool is added to the FarmBot system and recognized by the FarmBot app and the user can access and utilize the functionalities of the added tool.

Table 3.5: Add Tools Use Case for FarmBot

Use Case	Detect Weeds
Actors	User, Garden
Description	User initiates weed detection via the app. FarmBot captures images and gives them for analysis. User reviews images and identifications to manage weeds.
Preconditions	FarmBot operational, camera functional and app connected
Stimulus	User initiates weed detection in app.
Normal Flow	1. User defines settings. 2. User starts detection. 3. FarmBot captures images. 4. Images sent to app. 5. User reviews images and identifications. 6. User makes weed management decisions.
Exception Flow	Communication errors, image recognition limitations the user will be notified.
Post Conditions	User has images and potential identifications and informed for weed management.

Table 3.6: Detect Weeds Use Case for FarmBot

Use Case	Remove Weeds
Actors	User, Garden
Description	User selects removal method (manual or FarmBot-assisted) based on weed location. Initiates removal. User removes weeds manually or FarmBot removes based on location data.
Preconditions	FarmBot operational and weeds identified (manually or via app).
Stimulus	User initiates weed removal in app.
Normal Flow	1. User confirms weeds. 2. User defines weed locations (optional). 3. User initiates removal. 4. FarmBot removes based on location data. 5. User monitors and intervenes if needed.
Alternative Flow	4. User removes weeds manually.
Exception Flow	Communication errors and FarmBot limitations (reaching/removing weeds) sends error to user.
Post Conditions	Weeds removed

Table 3.7: Remove Weeds Use Case for FarmBot

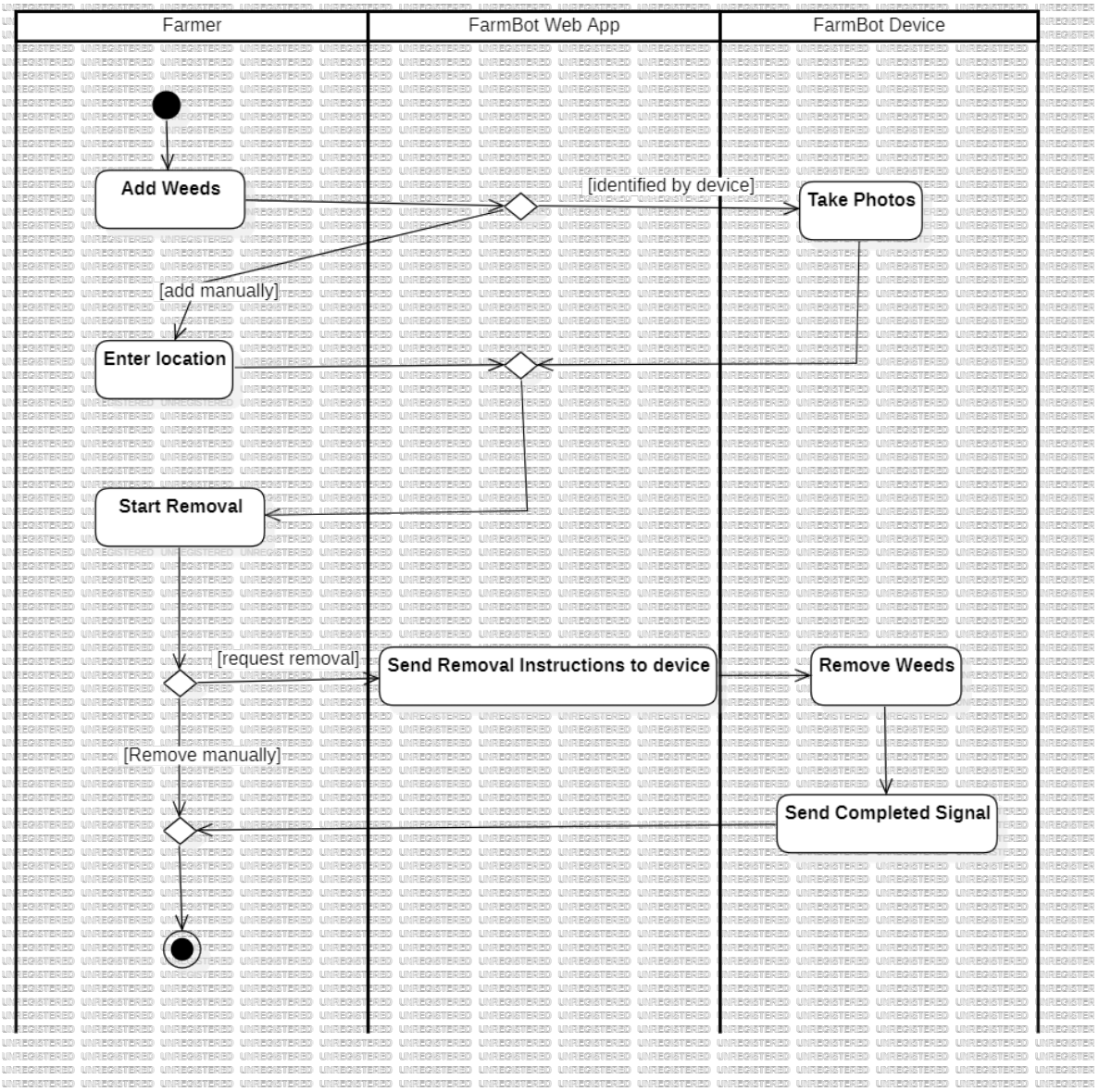

Figure 3.5: Activity Diagram for Remove Weeds Use Case

Use Case	Plant Crops
Actors	User, Garden
Description	User defines planting parameters in app, FarmBot follows plan, navigating and using seeder/planter to sow seeds.
Preconditions	FarmBot must be operational and plot must be prepared.
Stimulus	User initiates planting in app.
Normal Flow	1. User defines planting parameters (using plant data) 2. App generates planting plan. 3. User confirms plan and initiates planting. 4. FarmBot follows plan, navigating and using seeder/planter.
Exception Flow	Seeder malfunction/out of seeds, user must intervene.
Post Conditions	Seeds planted according to parameters, and user can monitor germination/growth.

Table 3.8: Plant Crops Use Case for FarmBot

Use Case	Get Plant Information
Actors	User, Data Sources, Gardem
Description	User searches for plant info in app. App retrieves data from external sources and displays it to the user.
Preconditions	App connected to internet.
Data Sources	User search terms and external plant databases (via app).
Stimulus	User searches for plant information in app.
Normal Flow	1. User searches for plant information. 2. App queries external databases based on search terms. 3. App displays retrieved plant information to user.
Exception Flow	Communication errors/limited info from external sources.
Post Conditions	User has access to plant information for decision-making.

Table 3.9: Get Plant Information Use Case for FarmBot

Use Case	Get Fertilizer Recommendation
Actors	User, Data Sources
Description	User provides plant data and receives fertilizer recommendations based on the information and external sources.
Preconditions	Data sources accessible and up-to-date
Data Sources	User input (plant type, soil conditions) and external fertilizer databases.
Stimulus	User initiates search for fertilizer recommendation.
Normal Flow	1. User enters plant data (e.g., type) and potentially soil conditions (optional). 2. External fertilizer databases are queried based on user input. 3. User receives recommended fertilizer types and application details.
Exceptions	Communication errors or limited data from external sources.
Post Conditions	User has access to fertilizer recommendations for their plant.

Table 3.10: Get Fertilizer Recommendation Use Case for FarmBot

3.3 Logical Database Requirements

a) **Types of Information Used by Various Functions**: The database should store information about plants, sensor data, tasks, users, and logs. This includes attributes such as plant name, species, sensor readings, task types, user credentials, and log descriptions.

b) **Frequency of Use**: Plant and sensor data may be accessed frequently for real-time monitoring. Tasks and logs may be accessed less frequently for reporting and analysis.

c) **Accessing Capabilities**: Users should have appropriate permissions to access, modify, and delete data based on their roles. Administrators may have full access, while regular users may have restricted access to certain data entities.

d) **Data Entities and Their Relationships**: Plants have sensor data and tasks associated with them. Users generate logs based on their actions. These re-

lationships should be represented in the database schema through proper table structures and foreign key relationships.

e) **Integrity Constraints**: Data integrity should be enforced through primary keys, foreign keys, and data validation rules. For example, each user may have a unique user ID, and tasks should be associated with valid plant IDs.

f) **Security**: Access to sensitive data should be restricted based on user roles. Passwords should be stored securely using encryption. Additionally, the database should have appropriate access controls to prevent unauthorized access and ensure data confidentiality.

g) **Data Retention Requirements**: Sensor data and logs may be retained for historical analysis, while user data should be retained for account management purposes. Regular backups should be performed to prevent data loss, and data retention policies should comply with relevant regulations and organizational policies.

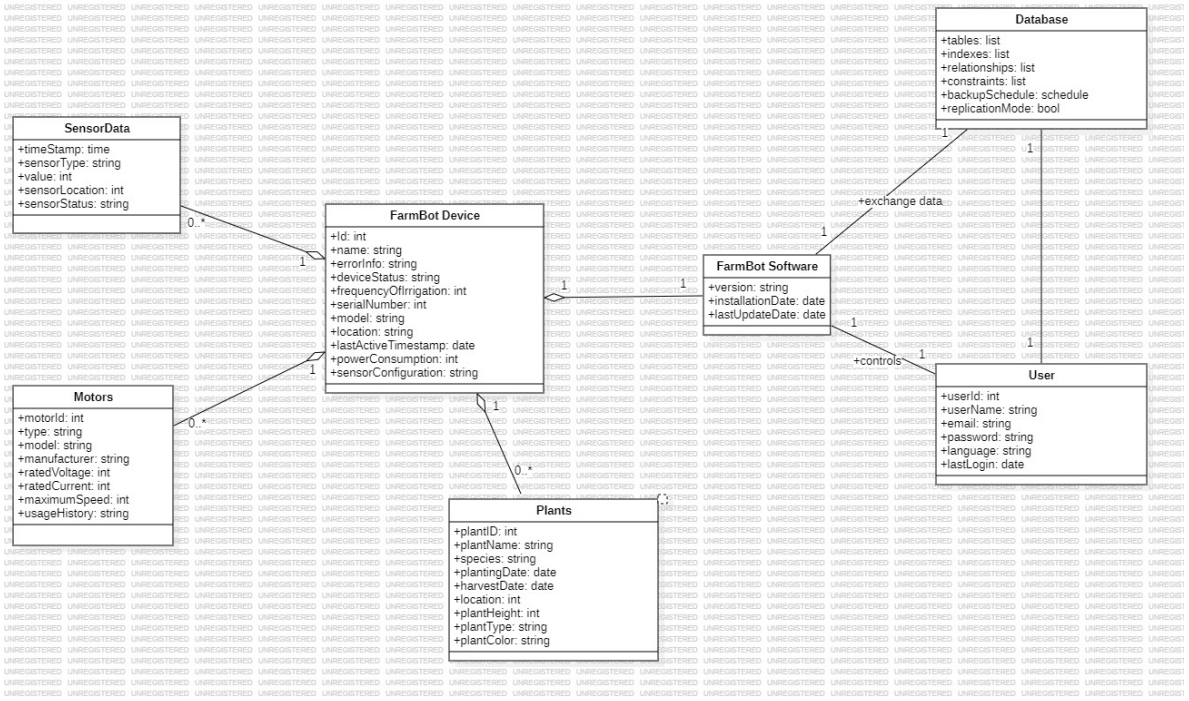

Figure 3.6: Logical Database Requirements Class Diagram

Refer to *(Clause 9.6.15)*.

3.4 Design Constraints

- To ensure safe functioning and reduce electrical dangers, FarmBot must follow electrical safety standards specified by organizations like UL (Underwriters Laboratories) or IEC (International Electrotechnical Commission).
- Materials used in FarmBot must be safe for the environment. Each component must be considered by its environmental impact.
- The web app of FarmBot should ensure that the personal information of its information collection/validation sources are protected.
- The website must contain security features, such as frequent security upgrades and data entry authentication, to guard against cyberattacks and illegal access

to data sources.

- Regulations may set forth criteria for food safety testing, crop traceability, and hygienic measures if FarmBot is used to grow food meant for sale or consumption.

3.5 System Quality Attributes

Here are some important quality attributes for the FarmBot system, listed in priority order with associated requirements:

3.5.1 Usability

FarmBot should be user-friendly and accessible to a broad audience, even those with limited technical experience.

- The FarmBot app interface (mobile/web) should be easy to understand and navigate. The interface must be user-friendly with clear menus and options.
- For setup, use, and troubleshooting, the app should include clear instructions and tutorials.
- The system should support multiple devices and input methods (e.g., touchscreen, voice commands).
- Users with disabilities should be able to access the system with features like text size adjustments and screen reader compatibility.

3.5.2 Reliability

FarmBot should function consistently and reliably to ensure the well-being of the plants it cares for.

- To reduce operational failures, FarmBot should be equipped with error detection and handling systems.

- The system should be designed to survive through potential harsh environmental factors (e.g., dust, moisture) that might disrupt operation.
- Regular software updates should be provided to address bugs and improve stability. These might be automatic or user-initiated software updates.

3.5.3 Performance

FarmBot should be able to complete duties effectively and without unwanted delays.

- The system should respond to user commands and provide feedback on time less than 2 seconds for basic actions .
- To prevent mistakes when planting, pulling weeds, or performing other duties, FarmBot's movements and actions need to be precise and controlled ± 1 cm is acceptable.
- To guarantee continuous operation, the system should optimize the use of resources..

3.5.4 Maintainability

FarmBot should be designed for easy maintenance and troubleshooting by users and/or support staff.

- The system should use well-documented and modular software components.
- Clear diagnostic tools and error messages should be given to help with troubleshooting.
- Clear user manuals and maintenance guides must be provided for hardware and software components.

3.5.5 Security

Security protocols need to be followed in order to protect user information and stop illegal access to the FarmBot system.

- If applicable, the transfer of data between the FarmBot software and the device must use secure communication methods.
- It is recommended to include user authentication measures in order to limit unapproved access to the FarmBot software and its features.
- The Farmbot system and software should be designed to be safe against potential cyberattacks or malware.

3.6 Supporting Information

FarmBot is an open source project that enables people to automate their gardening chores. Anyone can explore the functionality of the project and contribute to its development by downloading the FarmBot software and hardware designs for free from GitHub at https://github.com/FarmBot.

There are multiple ways to improve FarmBot. Developers can help with software development by introducing new features, addressing bugs, or enhancing current functionality. Another form of help could be gathering and verifying information about plant development, soil properties, and other farming aspects can yield significant benefits. The FarmBot community is centered around user input and experience sharing, rather than just coding and data. In addition to offering assistance to other FarmBot users, participation in forums and discussions can yield insightful information for future growth.

4. Suggestions to Improve The Existing System

4.1 System Perspective

The enhanced FarmBot system expands its reach with optional smartwatch control and eBay price lookups for informed resource management. It seamlessly integrates with AgriCloud Services for weather forecasts and soil analysis, while the web dashboard empowers farm managers with advanced analytics for data-driven decision-making.

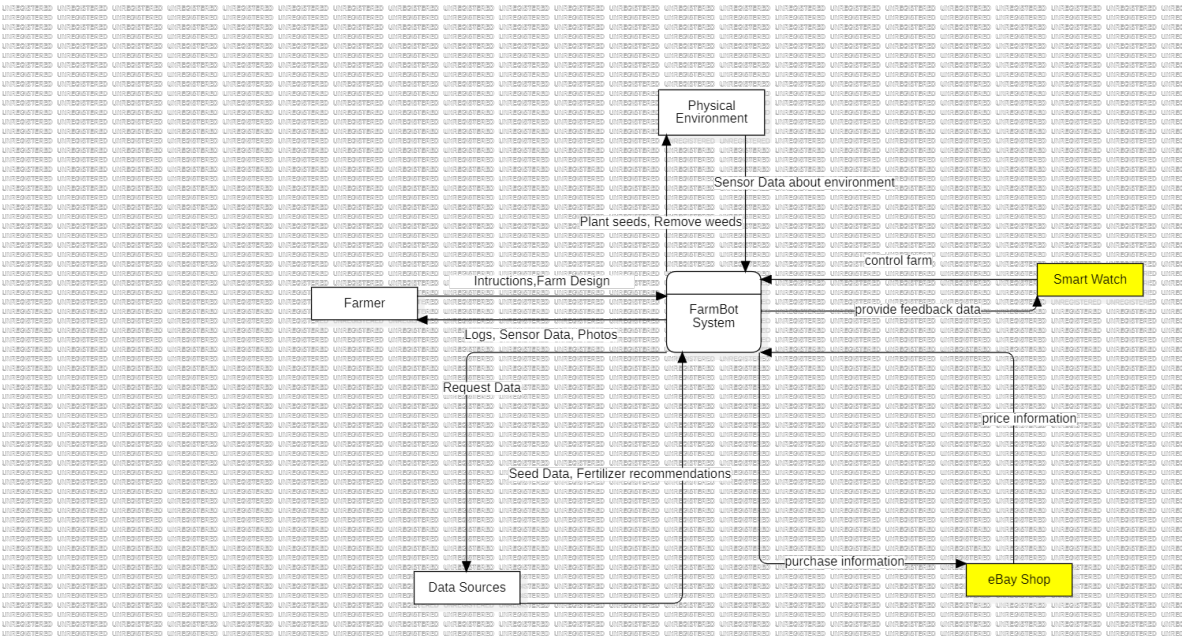

Figure 4.1: Context Diagram for Improved System

4.2 External Interfaces

Smart Watch Notification Center

The smart watch notification center enables users to command their FarmBot device, view weather information, check the number of plants currently being irrigated, and monitor various variables such as required fertilizer amount in real-time through an app on their smart watches. This greatly facilitates users' tasks and allows for instant notification to users in case of any errors, ensuring rapid information dissemination. Additionally, the ability for users to control the hardware from their smart watches is an exciting prospect. It represents a significant advancement in agricultural technology, empowering users with real-time insights, proactive notifications, and convenient control options. By leveraging the power of wearable technology, FarmBot users can streamline their farming operations, maximize efficiency, and cultivate thriving gardens with ease.

Community Interface

The community interface is essentially an online forum exclusively for FarmBot users. Here, users can seek solutions to any issues they encounter by posting their queries on the forum. The exclusivity of this forum to FarmBot users enhances the likelihood of finding solutions, as it increases the probability of multiple users experiencing similar issues. This targeted platform fosters a collaborative environment where users can effectively troubleshoot problems together.It is a private online discussion board intended only for FarmBot owners. Its goal is to encourage community problem-solving and collaboration by enabling users to ask for and offer support for any problems they run across with their FarmBot devices. By concentrating just on FarmBot users, the platform increases the possibility of discovering workable solutions and gives users the tools they need to solve obstacles collectively.

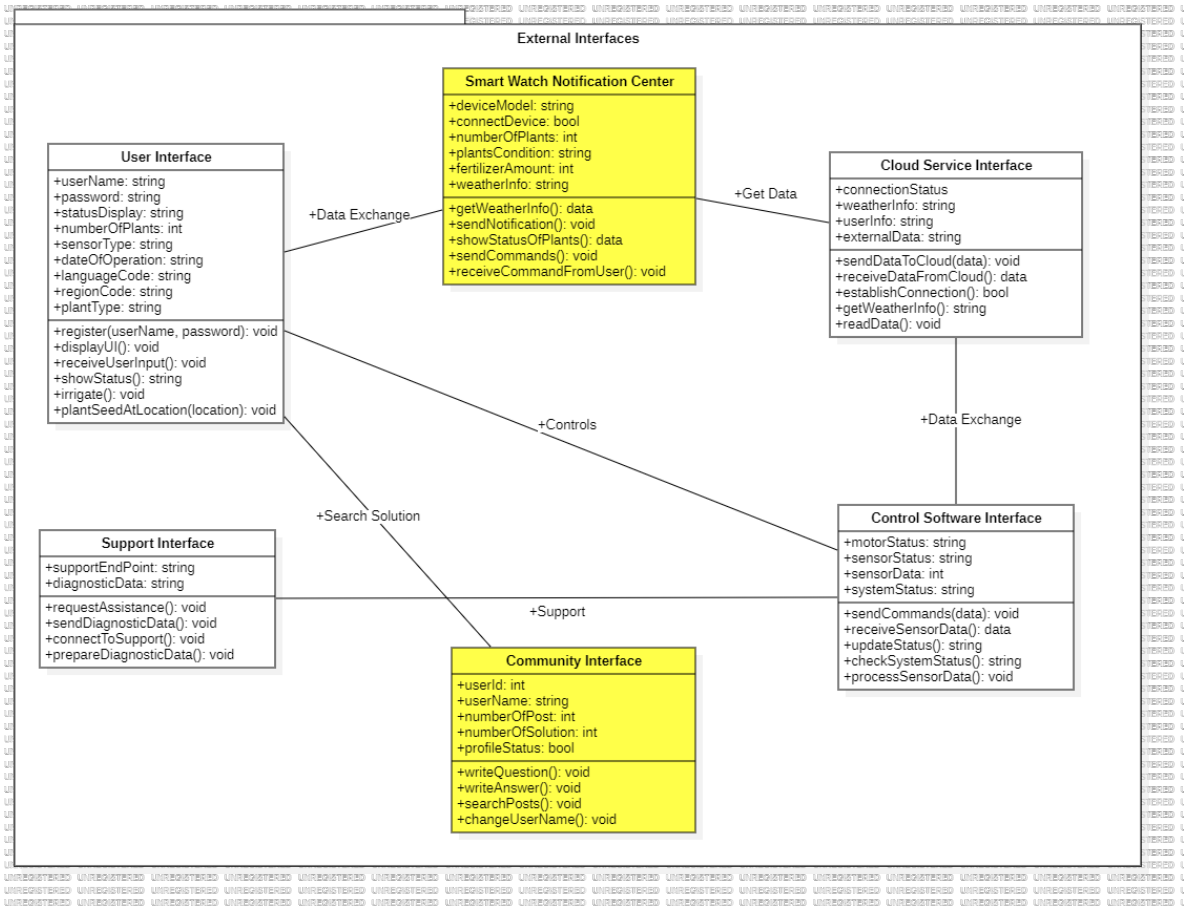

Figure 4.2: Class Diagram for External Interfaces Improved System

Refer to *(Clause 9.6.11, 9.5.8)*.

4.3 Functions

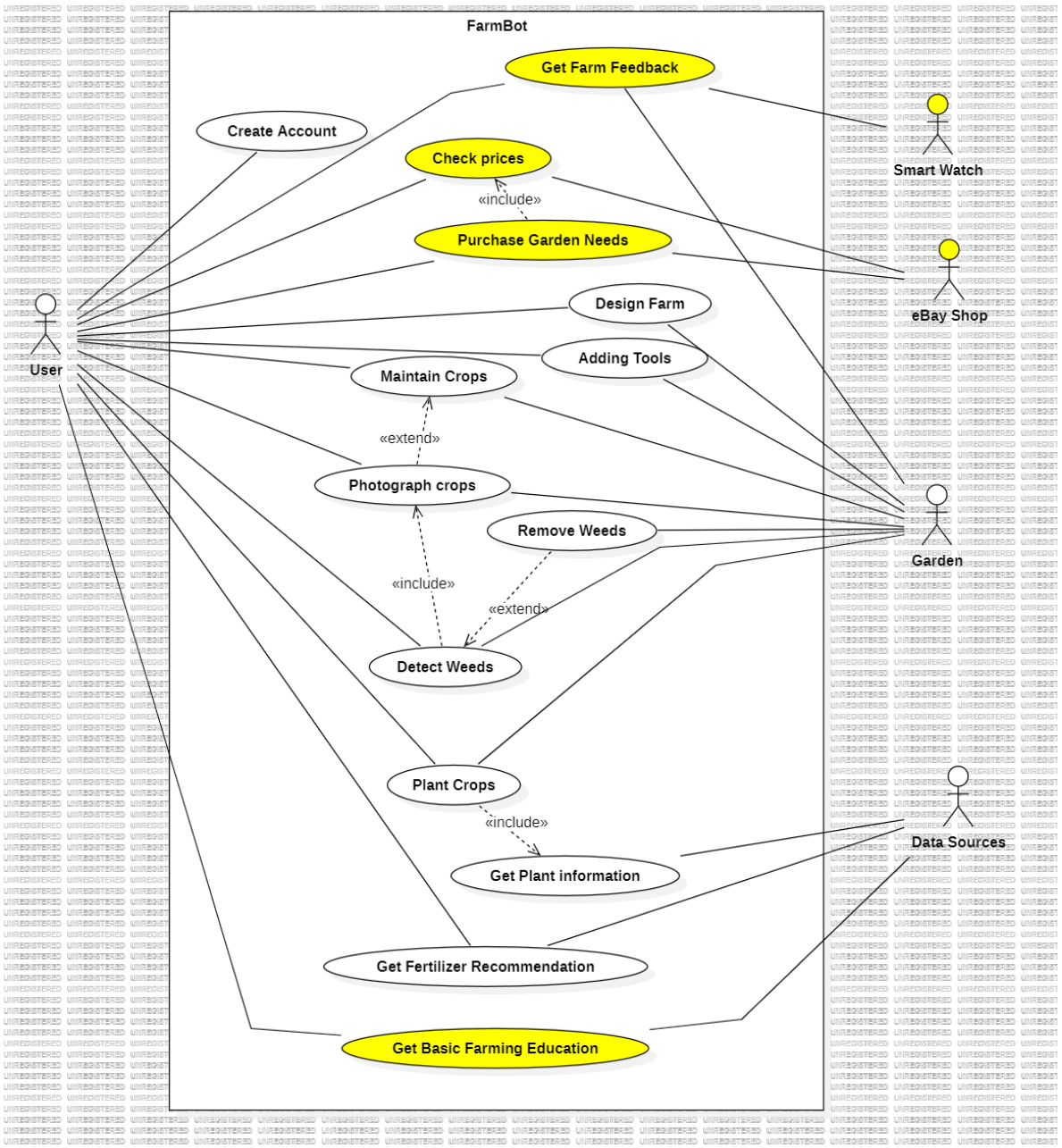

Figure 4.3: Improved Use Case Diagram

Use Case	Get Farm FeedBack
Actors	User, Smart Watch
Description	User checks information like garden state, planting state, sensor data etc. using the connection of his/her smartwatch to the device.
Preconditions	FarmBot operational and connectod to smartwatch.
Stimulus	User opens app in smartwatch.
Normal Flow	1. User opens app in smartwatch. 2. User requests specific information. 3. App gets data from Farmbot sensors etc. 4. User monitors and intervenes if needed.
Alternative Flow	3. App gets data from Internet.
Exception Flow	Communication errors and FarmBot limitations (reaching/removing weeds) sends error to user.
Post Conditions	Feedback received

Table 4.1: Get Farm Feedback Use Case for FarmBot

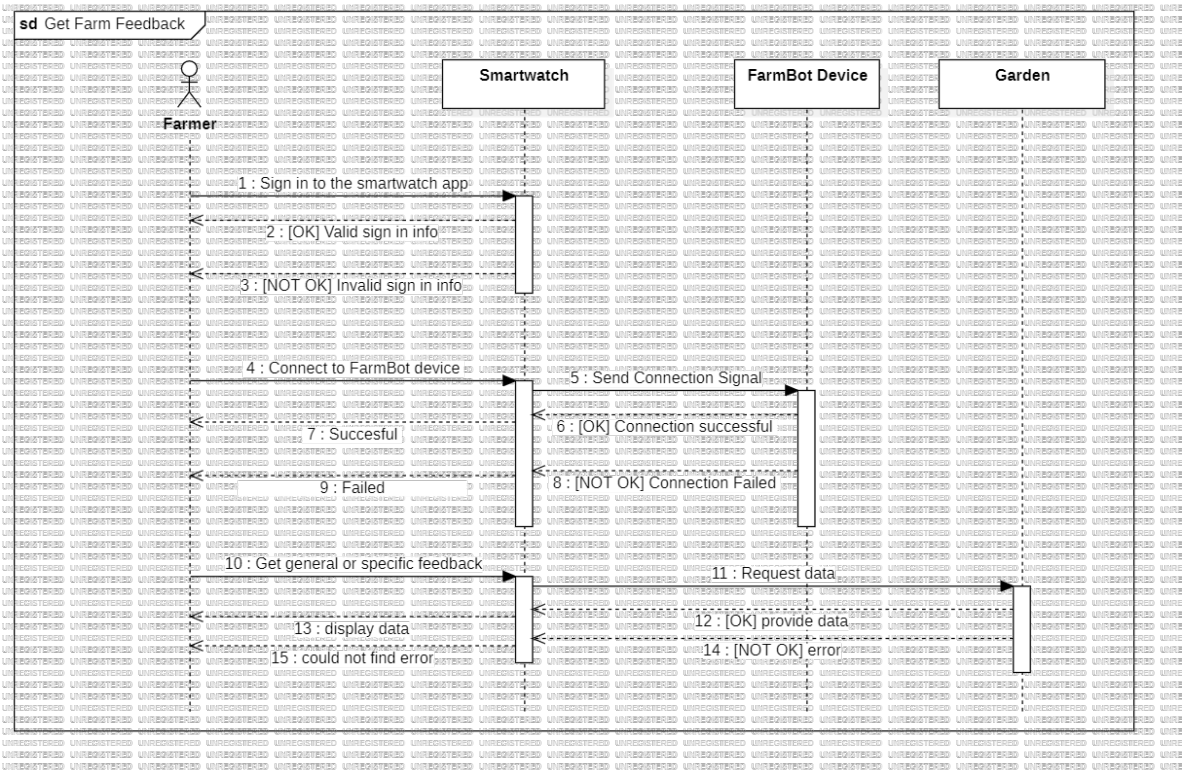

Figure 4.4: Get Farm Feedback Sequence Diagram

Use Case	Check Prices
Actors	User, eBay Shop
Description	User checks prices of things related to their farming needs such as seeds, fertilizers and tools.
Preconditions	Internet connection available, proper prompt given.
Stimulus	User requests price info in app.
Normal Flow	1. User opens FarmBot software. 2. User requests price information for the specified product. 3. App gets data for the specified product from eBay Shop. 4. User observes different prices from different sellers.
Exception Flow	Communication errors user is notified.
Post Conditions	Prices displayed to user.

Table 4.2: Check Prices Use Case for FarmBot

Use Case	Purchase Garden Needs
Actors	User, eBay Shop
Description	User checks prices of things they want to but and purchase them through the app.
Preconditions	Internet connection available, connected to eBay.
Stimulus	User requests a purchase.
Normal Flow	1. User opens FarmBot software. 2. User checks price information and request to purchase product. 3. App asks for payment information to provide to eBay. 4. App waits for approval from eBay.
Alternative Flow	2. User adds item to wishlist.
Exception Flow	Payment rejected user is notified.
Post Conditions	User purchased product.

Table 4.3: Purchase Garden Needs Use Case for FarmBot

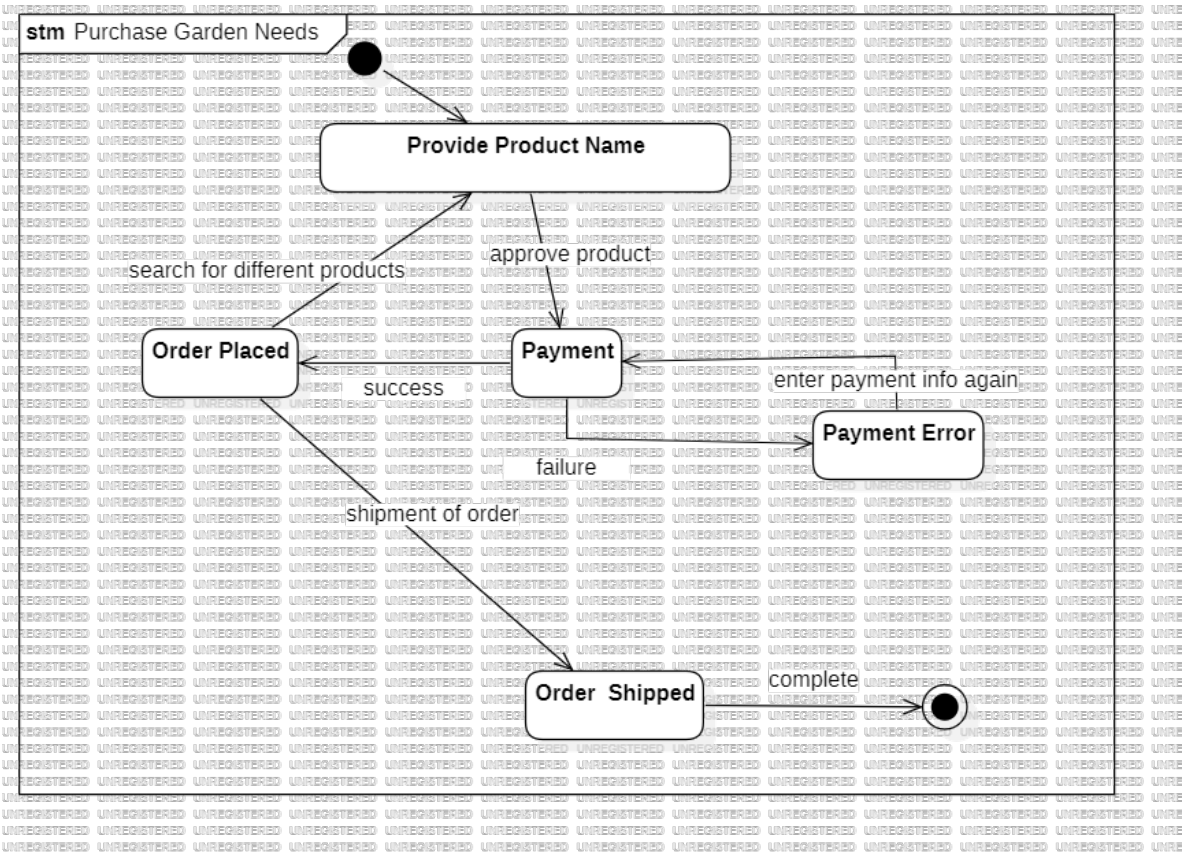

Figure 4.5: Purchase Garden Needs State Diagram

Use Case	Get Basic Farming Education
Actors	User, Data Sources
Description	User gets a bried education on farming (more specifically can get specific info on particular seeds etc.).
Preconditions	Internet connection available, Farmbot software connected to Data Sources.
Stimulus	User requests a certain education.
Normal Flow	1. User opens FarmBot software. 2. User searchs for specific education that they need. 3. Software gets information data from sources and provides to user.
Alternative Flow	2. User asks for a general education.
Exception Flow	Could not find specified education user receives error
Post Conditions	User took education.

Table 4.4: Get Basic Farming Education Use Case for FarmBot

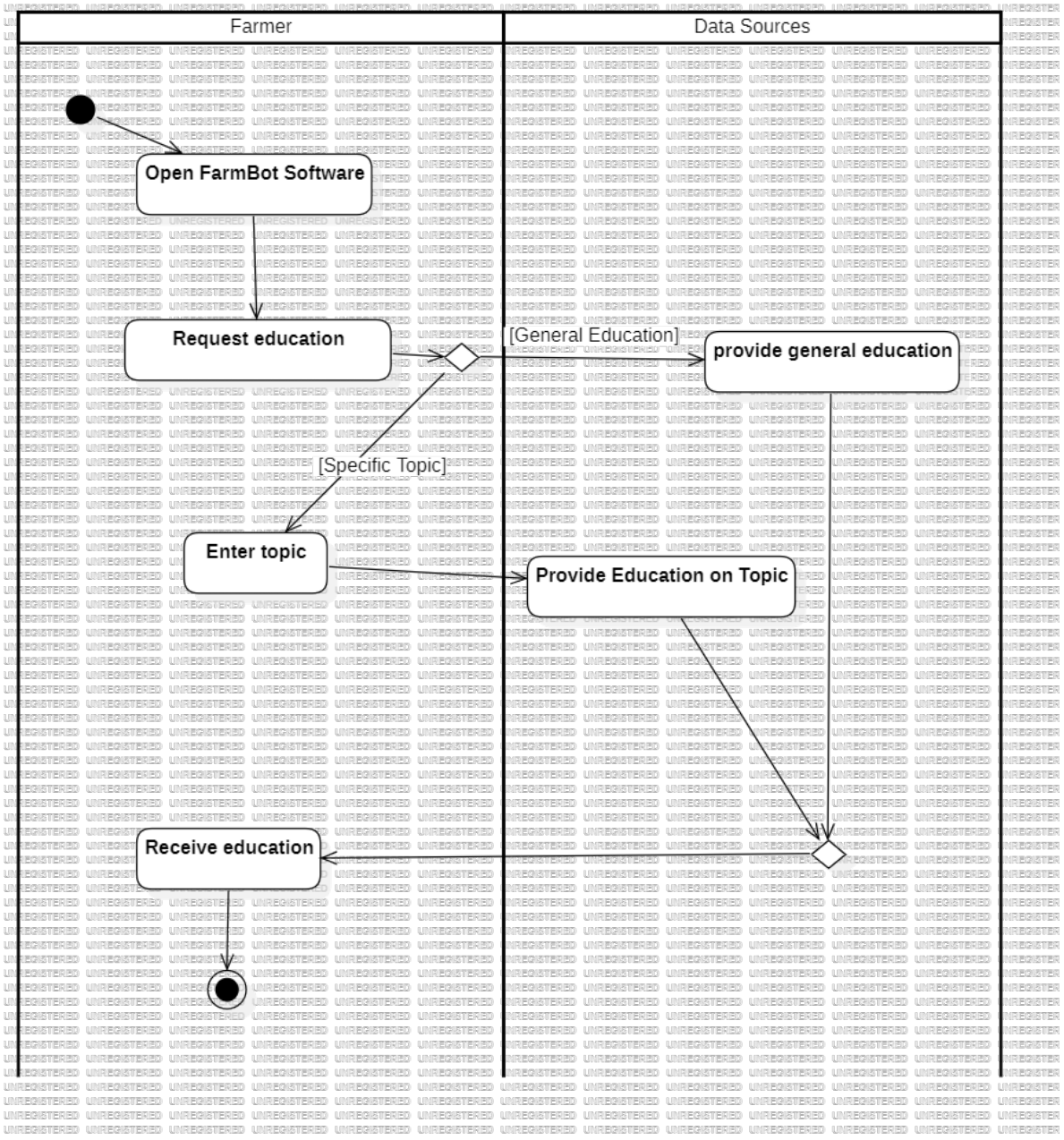

Figure 4.6: Activity Diagram for Remove Weeds Use Case

4.4 Logical Database Requirements

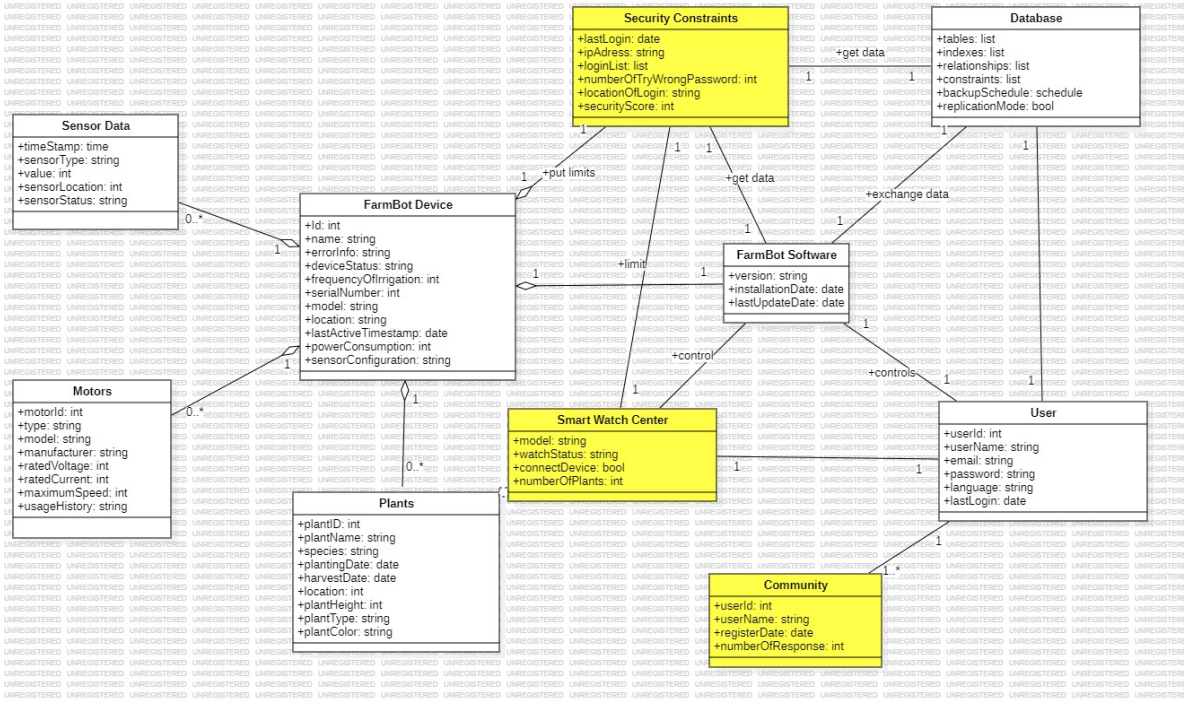

Figure 4.7: Logical Database Requirements Class Diagram Improved System

Security Constraints

One of the biggest vulnerabilities in the Farmbot system is the lack of security. Due to its open-source nature, every part of this system is already shared with the public, making it highly susceptible to misuse. Additionally, implementing a software that monitors user logins and tracks their activity can help assess the security score of each user. With this, limitations can be imposed on users with low security scores, especially during potentially risky operations. Since the tools used are essentially robots following commands, it's crucial to prevent misuse of these controls. Moreover, strong encryption protocols and multi-factor authentication systems can be established to greatly increase the security of the Farmbot system. Encrypting confidential information and demanding several levels of authentication before granting access to users can significantly lower the chance of unwanted access or data breaches. Additionally, frequent

penetration tests and security audits can assist in locating and addressing any possible weaknesses before bad actors can take advantage of them.

Smart Watch Center

The smart watch center serves as one of the key elements connecting the user to the Farmbot system. With this software, we can instantly receive data about Farmbot and respond to any issues promptly. Moreover, it's imperative for this device to adhere to security constraints, as it also serves as a control mechanism similar to the user interface. Therefore, they are closely associated with security constraints. Meanwhile, it ensures that necessary operations are carried out effectively with the user.

Community

Community has been integrated into the system to facilitate finding the necessary answers when encountering any issues. With only Farmbot users present in the community, along with individuals interested in this field, it enables communication and collaboration among peers. Additionally, users can seek solutions to others' problems within the community. Indeed, the significance of the community becomes even more pronounced when users face similar issues repeatedly.

4.5 Design Constraints

FarmBot's advancements must handle external constraints:

Regulations: Safety requirements may impose time limits or demand user presence. Water use and noise levels may be impacted by environmental regulations.

Technology: FarmBot's autonomy may be impacted by sensor capability and battery life. Cloud features may be restricted in remote regions due to limited internet availability.

Cost: Taking into account material costs and possible subscription fees, it's critical to strike a balance between functionality and affordability.

Usability: To prevent overwhelming consumers with technical intricacies, user inter-

faces must be simple to use. For setup or maintenance, you may still need to have some technical skills.

Lawful and moral: FarmBot must abide by data privacy laws if it gathers user data. Intellectual property rights must be respected when integrating with external services.

4.6 System Quality Attributes

Reliability:

- To ensure the FarmBot Web App's continued dependability and functionality under a variety of usage scenarios, extensive testing and maintenance procedures now include testing procedures for interactions with external components, such as the Community Forum, and eBay Shops.
- Implement robust error handling and recovery mechanisms.
- Ensure thorough testing and debugging procedures before delivery.
- Use reliable hardware components with low failure rates.
- Regularly update and maintain the software to address any potential reliability issues.

Availability:

- Make the necessary changes for Smart Watch Notification Center, eBay Shops, and Community Interface to comply with availability standards.
- Implement checkpoint mechanisms to save the system state and enable recovery in case of failure.
- Utilize automated recovery and restart procedures to minimize downtime.
- Design the system architecture with redundancy and failover capabilities to ensure continuous operation.

Security:

- User access restrictions for the Smart Watch Notification Center, Community Forum and e-Bay Shops are now included in authentication and authorization processes, guaranteeing safe access to data.
- Utilize cryptographic techniques to encrypt sensitive data and communication channels.
- Maintain detailed log and history datasets to track system activity and detect unauthorized access.
- Enforce strict communication protocols and firewall rules to restrict unauthorized access to the system.

Maintainability:

- Design the software with modular components to facilitate easier updates and maintenance.
- Use standardized interfaces and coding conventions to enhance code readability and maintainability.
- Implement version control and change management processes to track and manage software modifications on also Smart Notification Center effectively.
- Adapt the eBay Shops, Smart Notification Center, and Community Interface sections to these procedures to preserve the sustainability of the Farmbot.

Portability:

- Select a programming language or framework that has a reputation for being adaptable and interoperable with a variety of operating systems. This guarantees that the software can be installed on other systems with ease and doesn't require

a lot of work to modify. Such portable solutions include frameworks like.NET Core and languages like Java.

4.7 Supporting Information

With the improvements in place, FarmBot provides a more thorough and user-focused experience.

[2]