CENG350 SRS

Erencan Ceyhan
Alp Eren Yalçın
Group 62

April 9, 2024

Contents

List of Figures

List of Tables

1 Introduction

This document outlines the Software Requirements Specification (SRS) for an open-source initiative known as FarmBot [1]. FarmBot incorporates cutting-edge technology to enable individuals to grow plants in their own backyards, facilitated through an easy-to-use web application.

1.1 Purpose of the System

With the global population on the rise and an increase in food production to keep up, we're faced with a significant issue; the origins of our food are often a mystery, which isn't great for our health or the planet. Our current food production methods, relying on large-scale farms and factories, can't keep going forever. Enter FarmBot, an innovative fix that empowers individuals to cultivate their own plants and food at home, enhancing both oversight and clarity. Thanks to FarmBot, you can manage a compact, eco-friendly garden in your own space, leading to better food on your table and reducing dependence on the industrially produced variety.

1.2 Scope

FarmBot blends both software and hardware elements to offer users the ability to automate gardening tasks in their own spaces, with the added convenience of managing their FarmBot via a web application. Here's an overview of what the system covers:

- Offering a user-friendly web application that enables users to easily interact with the system.
- Encouraging developers to participate in and contribute to the FarmBot project.
- Facilitating users in customizing their virtual garden and engaging with the FarmBot device.
- Updating the physical garden to reflect modifications made in the web interface.
- Providing system updates to users from administrators.
- Alerting users about specific processes or events as they happen.

1.3 System Overview

1.3.1 System Perspective

The high level approach of the system can be defined as follows:

First FarmBot device is given a layout and configuration of the farm, which is provided by the customer. This configuration is then forwarded to the FarmBot's hardware components. Based on that, FarmBot provides necessities to the plants which is another component of the system, autonomously navigating through the farm, precisely delivering water, nutrients, and any required care to the plants. Simultaneously, the devices continually collects data on environmental conditions and plant health, ensuring optimal growth and yield. In this process device component provides the logs and other information also to the FarmBot software. As a last part, FarmBot is also showing to the end user, who is the customer, the logs and information about the plants and their status. In Figure 1 once can see the system context diagram of the FarmBot.

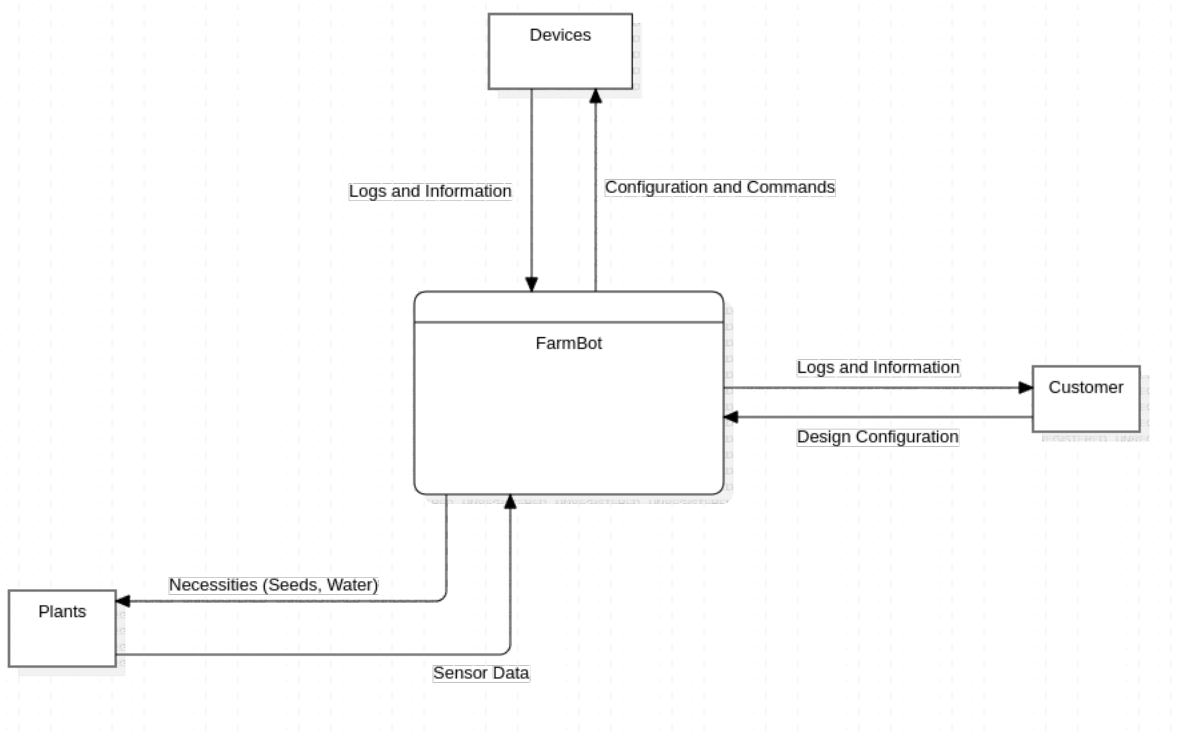

Figure 1: System Context Diagram

1.3.1.1 System Interfaces

FarmBot Express integrates various hardware and software components, which communicate through interfaces. These interfaces, which assist each other, include:

- User Interface

- Hardware Interface

- Software Interface

- Communication Interface

Initially, the user installs the device in their garden. Through the user interface, they can then manage the embedded operating system inside the FarmBot -known as "FarmBot OS"- in a high level and intuitive way. The hardware interface aids in verifying the device's basic setup correctness via sensors. Additionally, the website interacts with the device using the operating system through communication interfaces like HTTPS.

1.3.1.2 User Interfaces

Select Plants User Interface

This interface provides users with a library of plants to choose from, including detailed information about each plant's growth requirements and space needs. Users can:

- Browse through a selection of plant templates.
- Add plants to their digital garden by selecting a plant and clicking on the farm map.
- View plant-specific information, such as sowing depth, spread, and expected growth period.

FarmBot Farm Designer Map Interface The Farm Designer Map is a central feature that offers:

- A grid-based layout representing the physical garden space.
- Drag-and-drop functionality to place plants accurately within the grid.
- The ability to resize and rotate plants to reflect the actual garden setup.
- Visual indicators for plant life stages and water requirements.

Figure 2: Front-end Screenshot

Zoom In/Out and Layers Interface
This set of interface controls allows users to:

- Zoom in to view fine details or zoom out for a full-farm perspective.
- Toggle different layers on the farm map, such as plants, zones, soil moisture levels, and more.
- Customize the view for better planning and management of the garden.

Edit Plants User Interface
Once plants are added to the map, the Edit Plants UI enables users to:

- Update plant information as the plants grow and develop.
- Record planting and harvest dates, allowing the software to adjust care schedules.
- Modify the layout by moving, resizing, or deleting plant representations.

Group Plants User Interface
The Group Plants feature provides:

- A tool to create groups of plants, which can be managed together.
- The ability to apply sequences, schedules, and operations to the entire group.
- Custom groupings based on plant type, growth stage, or custom criteria set by the user.

Usability Considerations
The user interface is designed to be intuitive and easy to navigate for users at all levels of technical proficiency. Tooltips, contextual help, and a clean visual design guide the user through the application, making garden planning and management a straightforward and enjoyable experience.

Accessibility and Responsiveness
The interface adapts to various screen sizes, ensuring that users can access their Farmbot garden from a range of devices, including desktops, tablets, and smartphones. Accessibility features are in place to support users with disabilities.

1.3.1.3 Hardware Interfaces

The central piece of the hardware setup is known as "The Farmduino," a customized Arduino firmware tailored for FarmBot, which orchestrates the movement of motors and activation of various peripherals. The Raspberry Pi serves as the critical link for communication between the Farmduino and the FarmBot web application.

Users must install a specific operating system on the Raspberry Pi, dubbed "FarmBot OS," to ensure steady communication and coordination with the web app. This system also transmits instructions to the Farmduino, utilizing either a USB cable or a serial connection for communication. To get the Farmduino ready, users need to load FarmBot OS onto a Micro SD card using the Raspberry Pi Imager, which is then inserted into the Farmduino. This SD card comes pre-installed on the board. Upon starting up, FarmBot OS initiates a WiFi network for setup purposes. The inclusion of additional hardware components like sensors, encoders, and LEDs aids users in configuring the board effectively.

1.3.1.4 Software Interfaces

FarmBot OS maintains its link and stays in sync with the web application through a message broker. It also interacts with the physical components by issuing instructions to the Farmduino board and collecting data from various sensors and encoders.

On the server side, user authentication is handled using a JSON Web Token ystem. To protect against unauthorized access by potentially harmful software on client devices, Content Security Policies are employed [2]. The server's primary database is built on PostgreSQL, while Google Cloud Storage is utilized for storing files. Finally, the primary web framework used is Ruby on Rails.

1.3.1.5 Communication Interfaces

The system features various methods for connecting. The FarmBot API provides a REST API based on HTTP, designed for straightforward data exchange between the web browser and server using a request-response pattern. Within this setup, the message broker serves as a specialized component of the web API, using RabbitMQ for a unique style of communication that allows for instant message delivery without the need for direct requests, eliminating continuous message checks. For scenarios where the web app isn't operated within a browser, MQTT serves as the chosen protocol, linking the web browser with the message broker through WebSocket. Additionally, AMQP is employed by FarmBot OS for its communications with the web server.

1.3.1.6 Memory Constraints

Memory requirements aren't strictly defined, but for users looking to compile FarmBot OS from the source code, they'll need to ensure they have enough space on a 64-bit machine.

1.3.1.7 Operations

- **For the Unregistered User:**
  - New Account Creation and Verification
- **For the Registered User:**
  - Setup
  - Garden Design
  - Measure Soil Height
  - Regimen Creation
  - Event Schedule
  - Garden Monitoring
  - Scan for Weeds
  - Sequence Creation and Testing
- **For the Sensors and Cameras:**
  - Garden Monitoring
  - Scan for Weeds
  - Measure Soil Height
- **For the Plants:**
  - Garden Design
  - Regimen Creation
  - Event Schedule

1.3.2 System Functions

Use Case	Explanation
New User Account Creation and Verification	This functionality describes the steps for a new user to establish an account on the platform, from initiating account creation to completing email verification, to ensure authenticated access to services.
Setup Process	This functionality walks a user through the initial configuration of the FarmBot, from selecting the model to setting up WiFi and hardware, to attaching tools and sensors necessary for its operation.
Garden Design	This feature enables users to virtually lay out their garden using the application, allowing them to place plants and have these placements reflected in their actual garden for precise management.
Sequence Creation and Testing	This feature involves users in creating and testing command sequences that automate FarmBot operations, which can be stored for recurrent use, contributing to efficient garden handling.
Regimen Creation	This feature permits users to compile multiple sequences into a regimen that can be executed automatically, effectively using a predefined "recipe" over a plant's life cycle.
Event Schedule	This feature is about setting up automated tasks for the FarmBot to carry out at specific times, which can include scheduling sequences or regimens and defining their repetition and timing.
Garden Monitoring	This functionality allows users to observe their garden remotely by integrating a webcam feed into the FarmBot application, providing live visuals of the garden.
Scan for Weeds	This feature automates the detection of weeds by directing the FarmBot to take grid images and identify potential weed locations for subsequent action.
Measure Soil Height	This feature is used to gauge the level of the soil surface with FarmBot's cameras, crucial for precise sowing and maintenance tasks.

Table 1: Tabular description of use cases

1.3.3 Stakeholder Characteristics

End Users:
These individuals seek to leverage the benefits offered by FarmBot. They may lack the technical expertise to fully grasp the underlying mechanisms of FarmBot's operation. As such, the FarmBot web application is designed to be user-friendly, enabling operation without the need for technical knowledge.

Admins:
These individuals are tasked with ensuring the smooth operation of the website, guarding against any unforeseen issues.

Developers:
These participants contribute to the FarmBot project by developing software enhancements. As an open-source initiative, FarmBot welcomes developers to contribute to the improvement and customization of the project.

1.3.4 Limitations

- **Regulatory Compliance:** Usage of FarmBot must adhere to prevailing agricultural norms and local regulatory frameworks.
- **Hardware Constraints:** Constraints on FarmBot's hardware encompass electricity supply dependability, resistance to environmental conditions, adaptability to various terrains, lifting capabilities, precision of sensors, requirements for upkeep, reliance on network connectivity, and interoperability with ancillary hardware.
- **Application Integration:** Challenges in integrating FarmBot with external applications involve interoperability issues, restrictions on data exchange, and limitations in accessing external APIs.
- **Simultaneous Operations:** Limitations in running multiple FarmBot units simultaneously include restricted bandwidth for communications, potential scheduling conflicts, and the complexity of coordinating several devices.
- **Audit Trails:** Limitations in audit capabilities may prevent comprehensive monitoring and evaluation of agricultural activities, possibly leading to incomplete data capture.
- **Programming Language:** The system demands development in a high-level programming language that supports scalability and long-term enhancements. (Ruby)
- **Communication Protocols:** Essential communication protocols must be established to avoid delays and ensure the reliability of data transmission.
- **Quality Assurance:** Challenges related to the quality of FarmBot encompass the dependability and longevity of both hardware and software components.
- **System Criticality:** The application's critical nature involves ensuring hardware robustness, software dependability, and seamless connectivity.
- **Safety and Security:** These concerns revolve around preventing accidents, guaranteeing electrical safety, and securing sensitive information against breaches.
- **User Considerations:** Operators must maintain good physical health, possess sufficient technical expertise, and be environmentally conscious.
- **External Dependencies:** The system's performance is significantly reliant on continuous internet access for remote operations and receiving software updates.
- **Environmental Adaptability:** FarmBot must be capable of adapting to a wide range of environmental conditions, including extreme temperatures and varying levels of humidity, which might not always be feasible.
- **User Interface Design:** The complexity of the user interface may limit accessibility for users with varying levels of technical proficiency.

1.4 Definitions

Acronym	Description
API	An Application Programming Interface (API) is a protocol set enabling software applications to interact.
HTTP	Hypertext Transfer Protocol (HTTP) serves as the backbone for data transmission across the web.
OS	The operating system (OS) is the software that oversees computer hardware operations and supports software applications.
USB	Universal Serial Bus (USB) is a universal connection standard for attaching peripheral devices like keyboards and printers.
JSON	JavaScript Object Notation (JSON) is a format widely used for data exchange between web servers and applications.
PostgreSQL	An advanced, open-source relational database management system known for its robustness.
RabbitMQ	RabbitMQ is a free message broker software facilitating the exchange of messages among systems and applications.
MQTT	A compact messaging protocol, Message Queuing Telemetry Transport (MQTT) is designed for low-bandwidth, unreliable networks.
AMQP	Advanced Message Queuing Protocol (AMQP) ensures messaging interoperability across diverse systems.
SD Card	Secure Digital (SD) Card, a portable storage medium, is commonly utilized in digital gadgets for storage.

Table 2: Keywords and Their Descriptions

2 References

[1] FarmBot. FarmBot - GitHub. `https://github.com/FarmBot/Farmbot-Web-App`, 2024.

[2] FarmBot Developer Documentation. Responsible disclosure of security vulnerabilities. `https://developer.farm.bot/v15/other/responsible-disclosure-of-security-vulnerabilities`, 2024.

[3] FarmBot Documentation. Intro to FarmBot's software. `https://software.farm.bot/v15/docs/intro`, 2024.

[4] Anindita Roy Chowdhury and Anshu Gupta. Effect of music on plants – an overview. 4:30–34, 12 2015.

3 Specific Requirements

3.1 External Interfaces

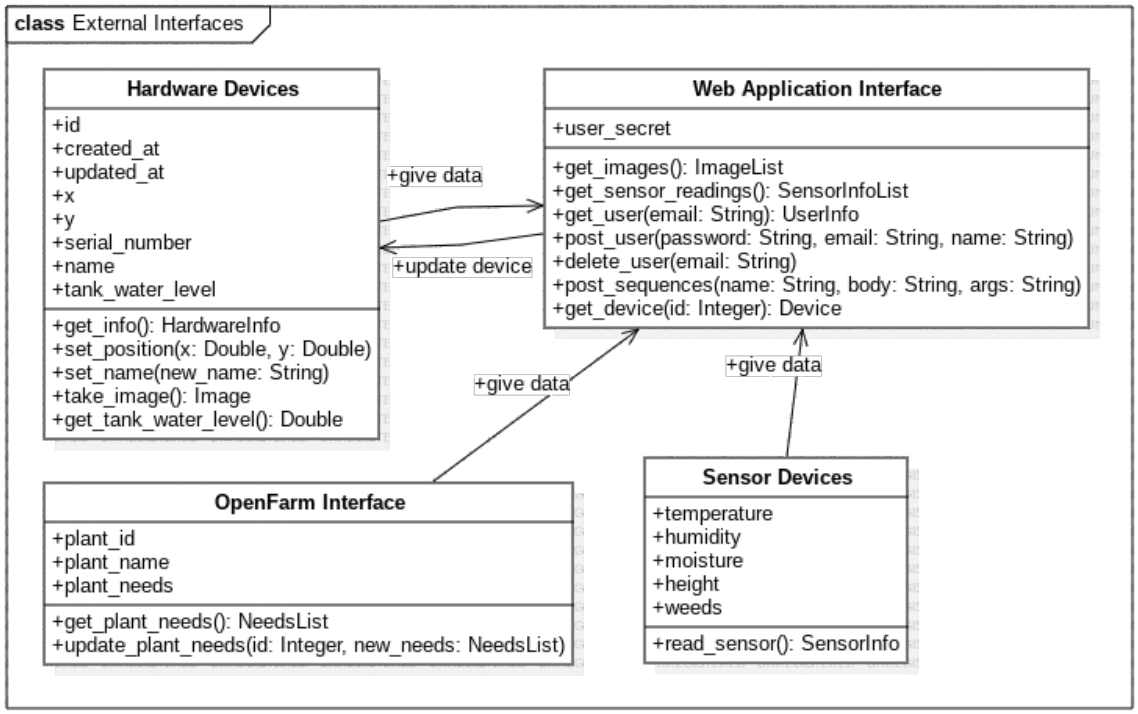

Figure 3: External Interfaces Class Diagram

3.2 Functions

[3]

The use-case diagram in Figure 4 includes 9 use cases and 5 actors. 2 actors on the left are different kinds of users with different roles: unregistered user and registered user. 3 actors on the right are the physical components of a typical FarmBot setup. The use-case diagram is additionally supported with description tables per use case.

Figure 5 has the Sequence Diagram for the "Measure Soil Height" use case. Figure 6 has the Activity Diagram for the "Create a Regimen" use case.

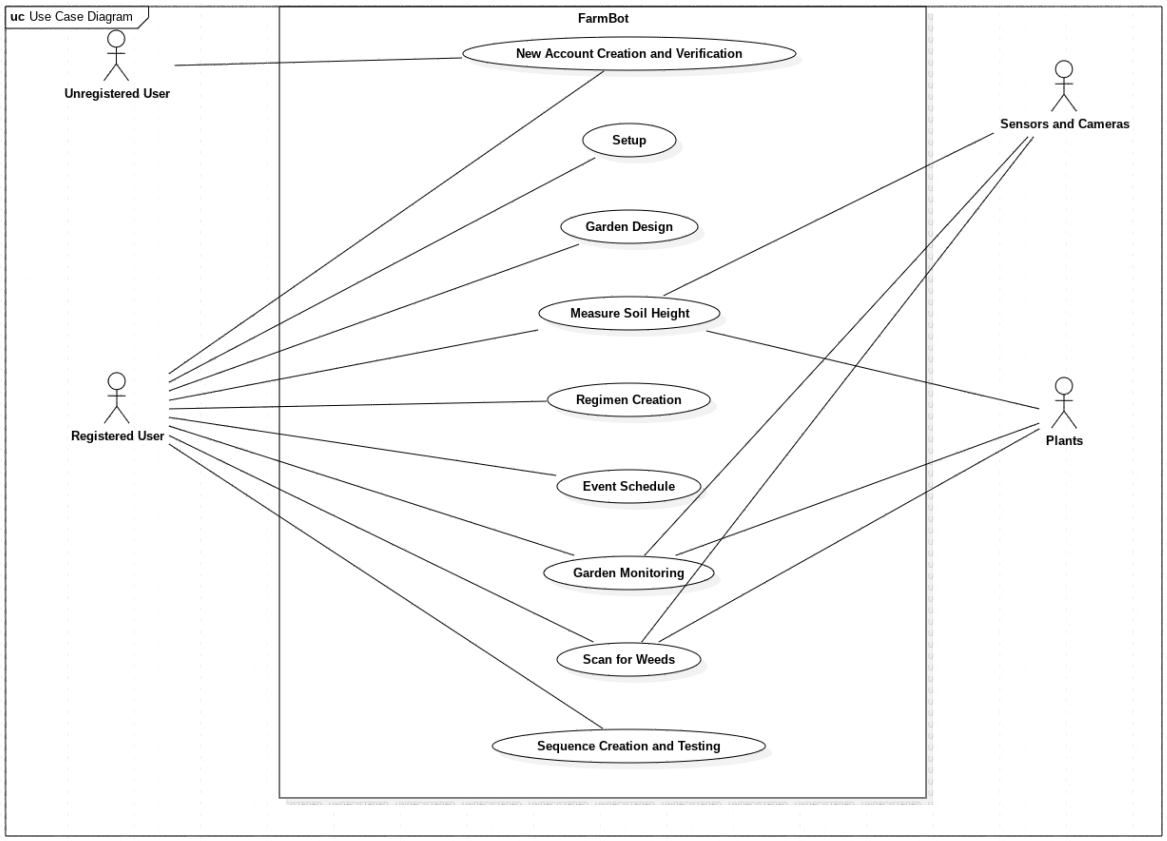

Figure 4: Use-Case Diagram

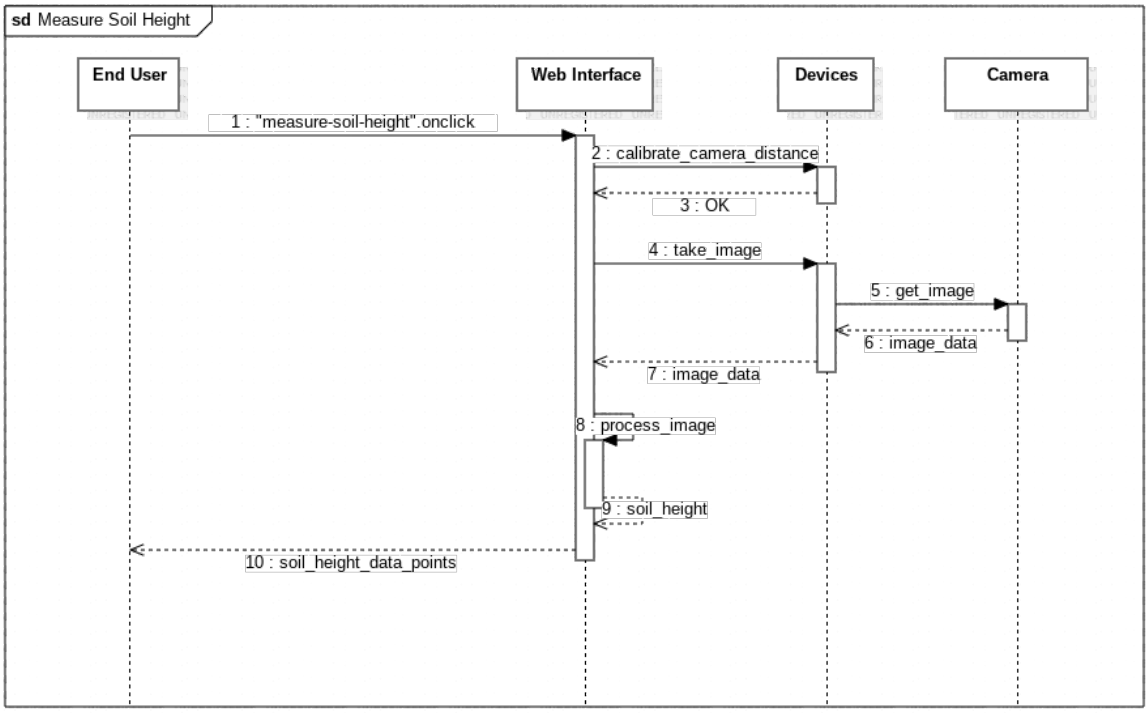

Figure 5: Sequence Diagram for Use Case **Measure Soil Height**

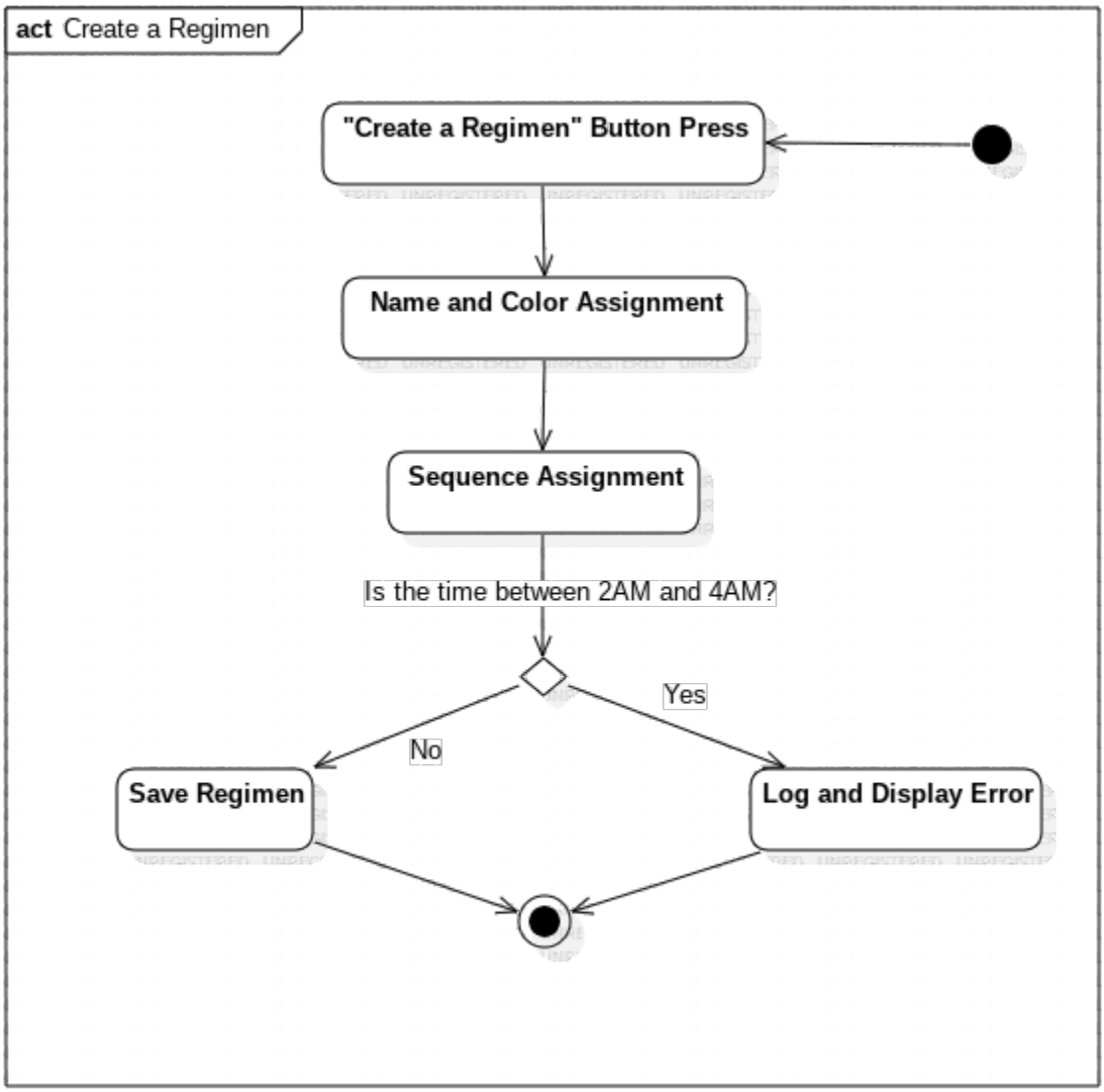

Figure 6: Activity Diagram for Use Case **Create a Regimen**

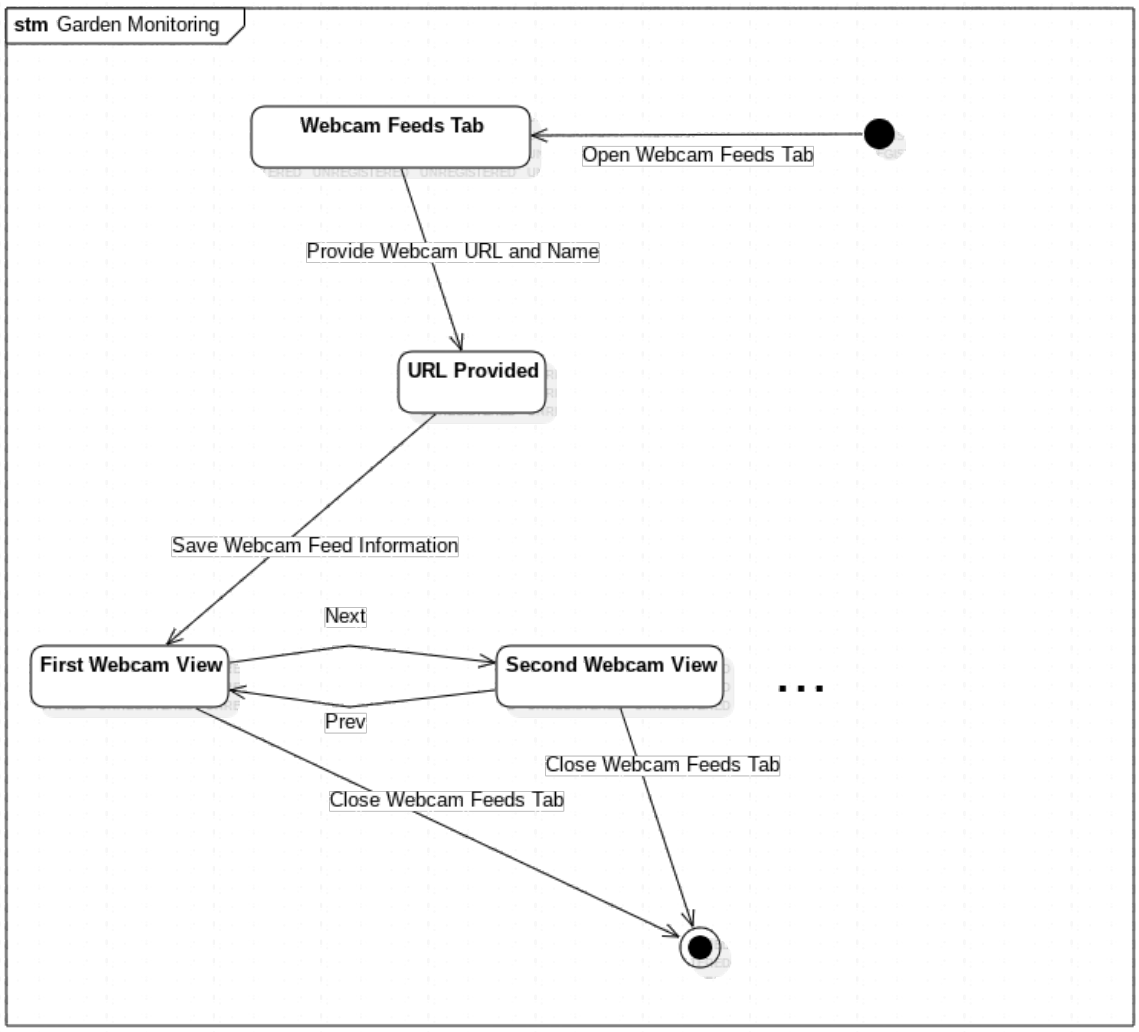

Figure 7: State Diagram for Use Case **Garden Monitoring**

Use Case Name	**New User Account Creation and Verification**
Actors	Unregistered User, Registered User
Description	This feature outlines the process for a new user to create an account on the web platform. It includes user interaction from account creation to email verification, ensuring the user can set up a verified account to access the system's services.
Pre-conditions	User must have access to a web browser and an internet connection.
Data	User's email address, password, and e-mail verification link.
Stimulus	User decides to create a new account.
Response	System sends a verification email.
Normal Flow	1. User navigates to my.farm.bot. 2. User enters required information and submits the form. 3. User checks email and clicks verification link.
Alternative Flow	-
Exceptional Flow	User does not receive verification email and requests a resend.
Post-conditions	User has an active, verified account and is logged in.
Comments	Ensure email system is reliable to prevent delays in account verification.

Table 3: Tabular description of the **New User Account Creation and Verification** use case

Use Case Name	**Setup Process**
Actors	Registered User
Description	This setup process guides the user through preparing the FarmBot for operation. It encompasses selecting the model, connecting to WiFi, configuring hardware parameters, and adding necessary tools and sensors for the FarmBot to function as intended.
Pre-conditions	User is logged in with a verified account.
Data	FarmBot model, order number, WiFi credentials, axis information.
Stimulus	User initiates the setup process.
Response	System guides user through setup steps, the real and virtual systems become correctly matched, system becomes ready to use.
Normal Flow	1. User provides their order number. 2. User selects their FarmBot model. 3. User connects to FarmBot. 4. User configures the virtual FarmBot's parameters to match them to the physical ones. 5. User adds tools, seed containers, slots, peripherals, and sensors.
Exceptional Flow	User encounters an issue and contacts support.
Alternative Flow	User skips entering order number, FarmBot model, virtual FarmBot parameter adjustment and/or extra peripheral additions.
Post-conditions	FarmBot is properly configured and connected.
Comments	Provide clear troubleshooting guides for common setup issues.

Table 4: Tabular description of the **Setup Process** use case

Use Case Name	**Garden Design**
Actors	Registered User
Description	The Garden Design feature allows users to plan and design their garden layout using the web application. Users can place plants virtually, and the system reflects these changes in the physical layout, enabling precise garden management.
Pre-conditions	FarmBot setup is complete.
Data	Plant types, locations, and planting instructions.
Stimulus	User wants to design their garden layout.
Response	System updates the virtual and physical garden layout.
Normal Flow	1. User adds and positions plants in the farm designer tool. 2. System saves the design.
Alternative Flow	-
Exceptional Flow	User designs an invalid layout.
Post-conditions	Garden design is saved and can be accessed or modified.
Comments	Include a library of common plants with default settings for ease of use.

Table 5: Tabular description of the **Garden Design** use case

Use Case Name	**Sequence Creation and Testing**
Actors	Registered User, Machinery
Description	Sequence Creation and Testing involve users automating operations by creating sequences of commands that the FarmBot executes. These sequences can be tested and saved for future use, aiding in efficient garden management.
Pre-conditions	FarmBot is configured; garden design is complete.
Data	Sequence commands, parameters.
Stimulus	User wants to automate a FarmBot operation.
Response	System executes and saves the sequence.
Normal Flow	1. User creates a new sequence. 2. User tests the sequence with the RUN button. 3. System saves the sequence if successful.
Alternative Flow	-
Exceptional Flow	Sequence does not perform as expected; user edits and retests.
Post-conditions	Sequence is available for scheduling or immediate execution.
Comments	Provide examples of common sequences to assist new users.

Table 6: Tabular description of the **Sequence Creation and Testing** use case

Use Case Name	**Regimen Creation**
Actors	Registered User
Description	The Regimen Creation function lets users compile a series of sequences into a regimen for automated execution. This feature makes it possible to use a "recipe" throughout the plant's lifetime.
Pre-conditions	User must have an account and be logged into the FarmBot web application. There should be existing sequences that the regimen can use.
Data	Sequences to include: A list of predefined actions (e.g., Seed Injection, Light Watering, Medium Watering) that the regimen will schedule. Timing for each sequence: Specific days and times when each sequence should run, relative to the start of the regimen.
Stimulus	The user initiates the creation of a new regimen by clicking on the appropriate UI element in the FarmBot web app.
Response	The system allows for the creation, editing, and deletion of regimens.
Normal Flow	1. User navigates to the Regimens panel and selects the option to create a new regimen. 2. User gives the regimen a descriptive name and optionally assigns it a color. 3. User schedules one or more sequences by specifying the sequence, time, and days for it to run. 4. User saves the regimen.
Alternative Flow	Editing a Regimen: If the user needs to edit an existing regimen, they can modify any of the sequences or timings and then save the changes. Deletion: A user can delete an existing regimen if it is no longer needed.
Exceptional Flow	User creates a regimen item between 2AM and 4AM.
Post-conditions	Once created or edited, the regimen is stored and available for scheduling through events. Changes are reflected in all instances where the regimen is scheduled.
Comments	Regimens are designed to make the scheduling of sequences for plant care more efficient, allowing for reuse and easy adjustment to schedules. To effectively run a regimen, it must be scheduled via an event with a specific start date.

Table 7: Tabular description of the **Regimen Creation** use case

Use Case Name	**Event Schedule**
Actors	Registered User
Description	Event Scheduling is the process of setting up tasks for FarmBot to perform at predetermined times. Users can schedule sequences or regimens to execute, with options to repeat these events and specify start and end times.
Pre-conditions	The user must be logged into the FarmBot web app and be on the Farm Designer page.
Data	User inputs include sequence or regimen to execute, start date and time, and optionally, repeat interval and stop date and time. If the sequence or regimen includes externally defined variables, values for these must also be provided.
Stimulus	The user navigates to the events panel and chooses to create a new event.
Response	The system schedules FarmBot actions based on the provided inputs.
Normal Flow	User creates an event by selecting a sequence or regimen, setting start and optionally end times, and saves the event.
Alternative Flow	Editing an existing event to change its details.
Exceptional Flow	Attempting to schedule events during FarmBot update times (2AM to 4AM) should be avoided to prevent conflicts. Changing between a sequence and a regimen.
Post-conditions	The event is saved and displayed in the agenda, ready to be executed by FarmBot at the scheduled times.
Comments	Events must be scheduled at least one minute into the future to ensure proper execution.

Table 8: Tabular description of the **Event Schedule** use case

Use Case Name	**Garden Monitoring**
Actors	Registered User, Sensors and Cameras, Plants
Description	Garden Monitoring enables users to keep an eye on their gardens from afar. By putting a webcam feed into the FarmBot web application, users can view live images to monitor the garden remotely.
Pre-conditions	User must have a FarmBot setup and connected to the internet. An external camera system compatible with FarmBot's software, such as a Nest home security camera, must be available.
Data	Name of the webcam feed. Publicly accessible URL or IP address of the webcam stream.
Stimulus	User wants to monitor their garden remotely through webcam feeds.
Response	Webcam feeds are displayed within the FarmBot web application, allowing remote monitoring.
Normal Flow	User accesses the Webcam Feeds tab. User adds a new webcam feed by providing a name and URL/IP address. User saves the webcam feed information. The webcam feed is available for viewing within the FarmBot web application.
Alternative Flow	If multiple webcam feeds are added, the user can cycle through them using "PREV" and "NEXT" buttons.
Exceptional Flow	If a webcam service does not allow embedding into other sites or apps, the user must contact the webcam's customer support for solutions.
Post-conditions	The user can view the garden remotely through the added webcam feeds. Webcam feeds can be deleted or edited as needed.
Comments	The system supports adding multiple webcam feeds for comprehensive monitoring. Some webcam services' security policies may prevent embedding, requiring user action outside the FarmBot software to resolve.

Table 9: Tabular description of the **Garden Monitoring** use case

Use Case Name	**Scan for Weeds**
Actors	Registered User, Sensors and Cameras, Plants
Description	The Scan for Weeds feature automates the process of detecting weeds in the garden by utilizing the camera in a FarmBot setup. The system guides the FarmBot to take pictures of the grid and detecting potential weed locations for further treatment.
Pre-conditions	FarmBot's camera is calibrated. The user is logged into the FarmBot Web App.
Data	A grid of points representing each location for photo capture in the garden. A color assigned to these points for easy grouping. Two sequences: one for moving to a point and detecting weeds, and another for executing the first sequence across all points.
Stimulus	The user initiates the "Scan entire garden for weeds" sequence.
Response	FarmBot moves to each point in the grid and performs weed detection.
Normal Flow	Create a point grid in the garden area where photos will be taken. Group these points by the assigned color. Create a sequence for detecting weeds at a single point. Create a sequence to execute the weed detection sequence at all points in the group. Run the "Scan entire garden for weeds" sequence.
Alternative Flow	Adjust the point grid's locations, spacing, or number of points based on initial results. Modify weed detection parameters for better accuracy.
Exceptional Flow	Camera calibration issues prevent accurate photo capture or weed detection. Sequence execution failure due to incorrect sequence configuration or hardware issues.
Post-conditions	The garden has been scanned for weeds, with potential weed locations identified. Adjustments may be needed for point grid setup or weed detection parameters.
Comments	This process may require trial and error to optimize the spacing and number of points for your specific garden setup. Adjusting weed detection parameters can significantly affect the accuracy of the weed identification process.

Table 10: Tabular description of the **Scan for Weeds** use case

Use Case Name	**Measure Soil Height**
Actors	Registered User, Sensors and Cameras, Plants
Description	Measure Soil Height is used to determine the soil surface level in the garden with the calibrated cameras on a FarmBot setup, which capture images and processes them to find soil height, which is crucial for various tasks such as precise planting and maintenance.
Pre-conditions	The FarmBot is assembled, connected, and operational. The camera is installed on the FarmBot and properly calibrated. The FarmBot Web App is accessible, and the user is logged in.
Data	Distance from the camera lens to the soil surface in millimeters. Soil height points collected from various garden locations. Images captured for measuring soil height.
Stimulus	The user initiates the soil height measurement via the FarmBot Web App or through a pre-defined sequence that includes the Measure Soil Height command.
Response	Processed images indicating soil height. Soil height data points generated and stored in the system.
Normal Flow	Calibration of the camera's distance from the soil. Capturing images at specified locations within the garden. Processing images to determine soil height. Recording soil height data points.
Alternative Flow	Manual addition of soil height points for comparison or in case of measurement issues.
Exceptional Flow	Errors in image processing due to poor lighting or obstruction. Calibration errors requiring reset and re-calibration.
Post-conditions	Soil height data accurately reflects the surface level of the garden's soil. The FarmBot is ready for subsequent gardening tasks, utilizing the soil height data for precise operations.
Comments	Ensure smooth movement of Y and Z axes and clear visibility of soil for effective measurement. Regular updates and calibration checks are recommended for accuracy.

Table 11: Tabular description of the **Measure Soil Height** use case

3.3 Logical Database Requirements

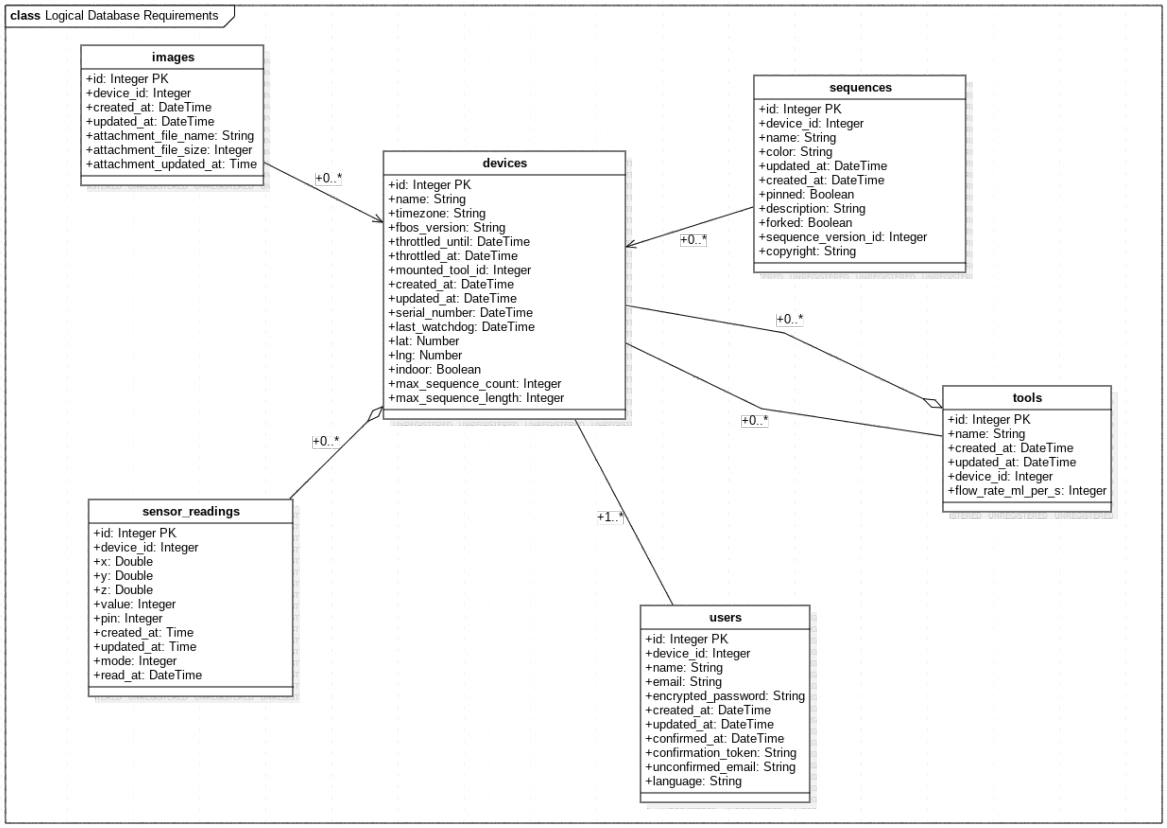

Figure 8: Logical Database Requirements Diagram

- **Images**: Images are taken by devices, with each image having a unique identifier and metadata including the device ID, creation timestamp, and file size.
- **Devices**: Devices are central to the system, each with a unique name and version. They track events through timestamps and have a set number of maximum sequences they can perform.
- **Sensor Readings**: Devices record sensor readings that include location coordinates and timestamps, which are essential for monitoring farm conditions.
- **Sequences**: Sequences represent automated processes or routines that devices can execute. These sequences have properties such as color and description, and are versioned to track updates.
- **Tools and Usage**: Tools, which are utilized by the devices for various farm activities, are cataloged with details like flow rate, tied to a specific device.
- **Users**: Actual endpoints that are using the device. Stored information includes things such as name, email and etc.

3.4 Design Constraints

- FarmBot is dedicated to democratizing farming through technology, offering its platform as entirely open source for full transparency and community engagement.

- While contributions are welcomed, developers are advised to discreetly report any security issues directly to the team via email due to sensitivity concerns.

- Access to FarmBot is restricted to individuals who are at least 13 years old, ensuring the system's use is appropriate and safe.

- The usage of FarmBot must align with the legal and regulatory confines of California, U.S.A, ensuring compliance with local laws.

- FarmBot nurtures a worldwide community, driving innovation and growth by pooling collective insights and developments.

- Emphasizing user privacy, FarmBot adopts strict protocols to safeguard personal data against unauthorized access.

- Integration with a variety of gardening tools enhances the FarmBot experience, making personal agriculture more efficient and enjoyable.

- Regular updates to FarmBot software guarantee compatibility with cutting-edge gardening techniques, ensuring users have access to the best possible tools.

3.5 System Attributes

- **Consistency:** Executes regular tasks such as irrigation, seeding, and soil health assessment with minimal interruptions.
- **Operational Readiness:** Accessible for use at any time, characterized by minimal operational pauses and swift error resolution.
- **Data Protection:** Ensures the safety of information and operational controls against unauthorized modifications or access.
- **Efficiency:** Conducts agricultural operations promptly and efficiently, optimizing the deployment of resources.
- **Flexibility:** Can easily accommodate changes or upgrades in technology and farming practices without significant overhauls.
- **Resilience:** Possesses the ability to withstand and quickly recover from natural adversities or technical malfunctions.
- **Customizability:** Offers a range of options that can be tailored to meet the specific needs and preferences of different farming operations.

3.6 Supporting Information

Farm.bot combines technology with gardening through an open-source platform that automates plant cultivation. Accessible on GitHub, it encourages developers to contribute, continuously enhancing efficient and sustainable home gardening.

4 Suggestions to Improve the Existing System

4.1 System Perspective

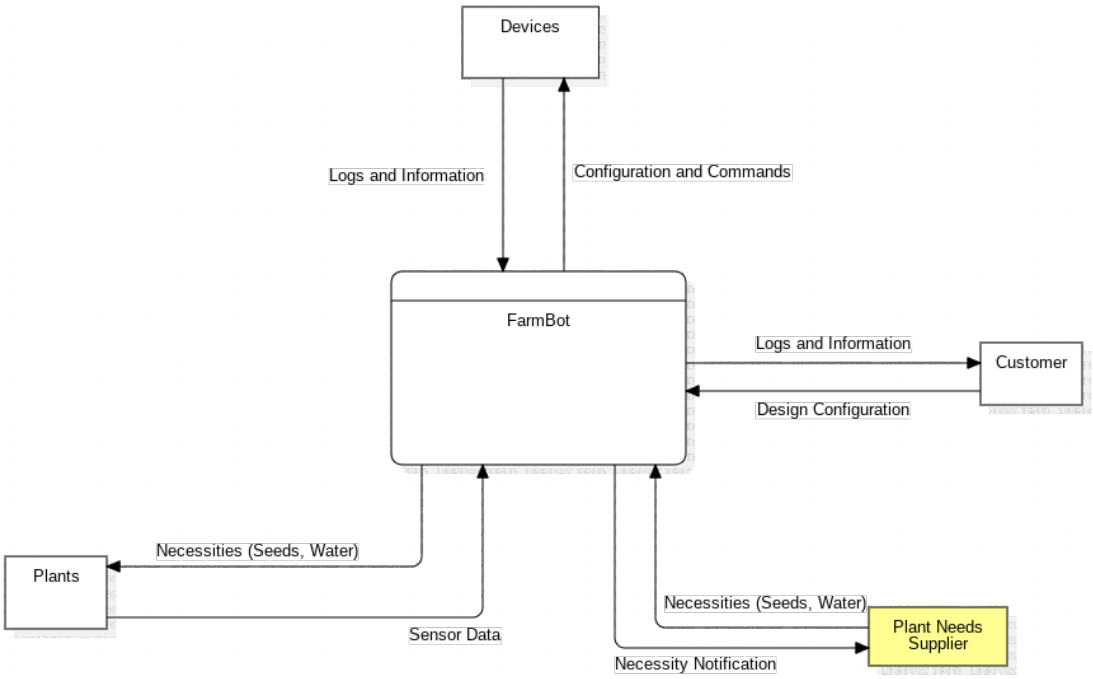

Figure 9: System Context Diagram for Improved FarmBot

We have one more component in the System Context Diagram for Improved FarmBot, which is Plant Needs supplier. This component mostly corresponds to our suggestion: "Plant Needs Delivered by Drone Delivery / Buy needs with Drone Delivery".

This feature allows a registered user to purchase items online and have them delivered to their location via a drone.

Briefly our 4 suggestions are:

- Create Timelapse Video
- Buy Needs with Drone Delivery
- Store Water from Rain
- Make Plants Listen to Music

4.2 External Interfaces

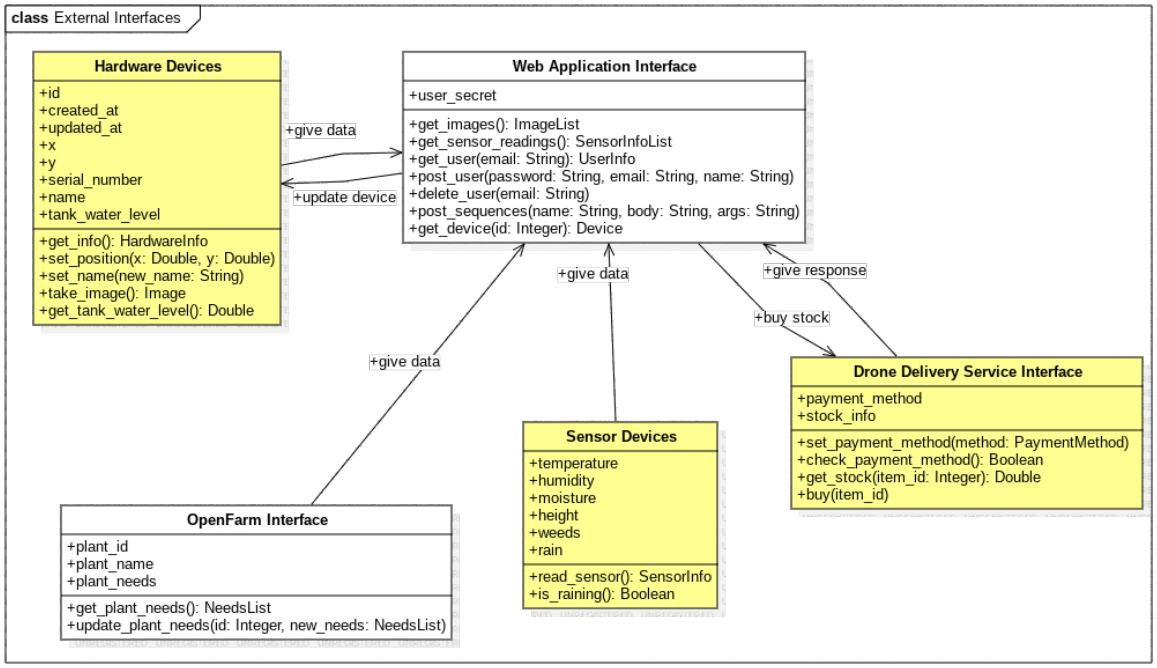

Figure 10: External Interfaces Class Diagram for Improved FarmBot

- **Web Application Interface Enhancements:**
  - `+post_timelapse(request): Video` – Command to generate a timelapse video of plant growth.
  - `+order_delivery(details): Status` – Function to place orders for garden needs with drone delivery.
  - `+store_water(command): Status` – Feature to store rainwater based on system commands.
  - `+play_music(playlist): Status` – Ability to play selected music for the benefit of plant growth.
- **Drone Delivery Service Interface Additions:**
  - `+payment_method` – Field for specifying payment method.
  - `+set_payment_method(method)` – Method to set a payment method for drone deliveries.
  - `+stock_info` – Field likely containing stock information for items deliverable by drone.
  - `+get_stock_item(id): Double` – Retrieves stock data for an item, such as price or quantity.
  - `+buy(item_id)` – Initiates the purchase process for drone delivery.
- **Sensor Devices Update:**
  - `+is_raining(): Boolean` – Method to detect rain, potentially for automating water collection.

4.3 Functions

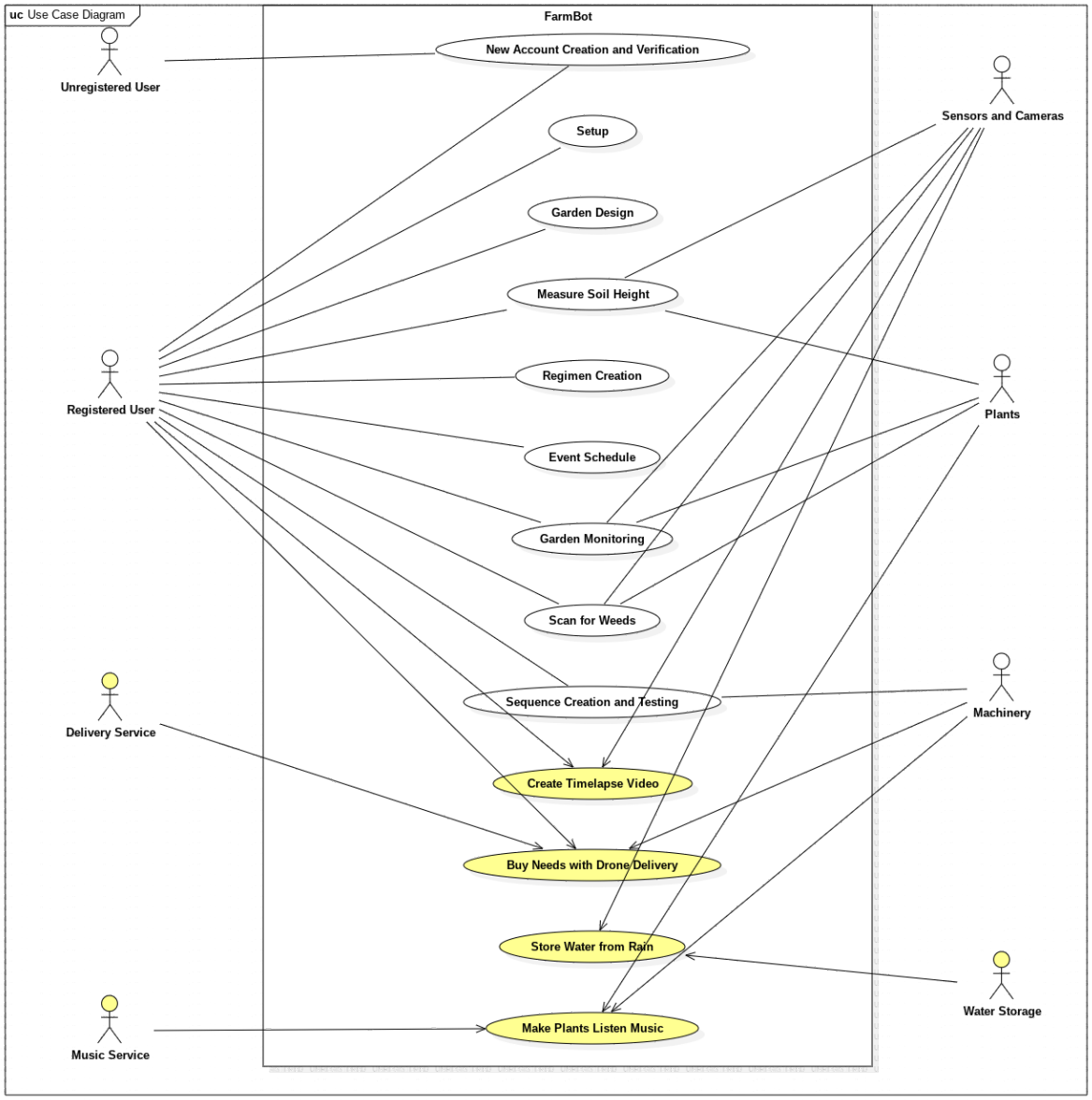

Figure 11: Use-Case Diagram for Improved FarmBot

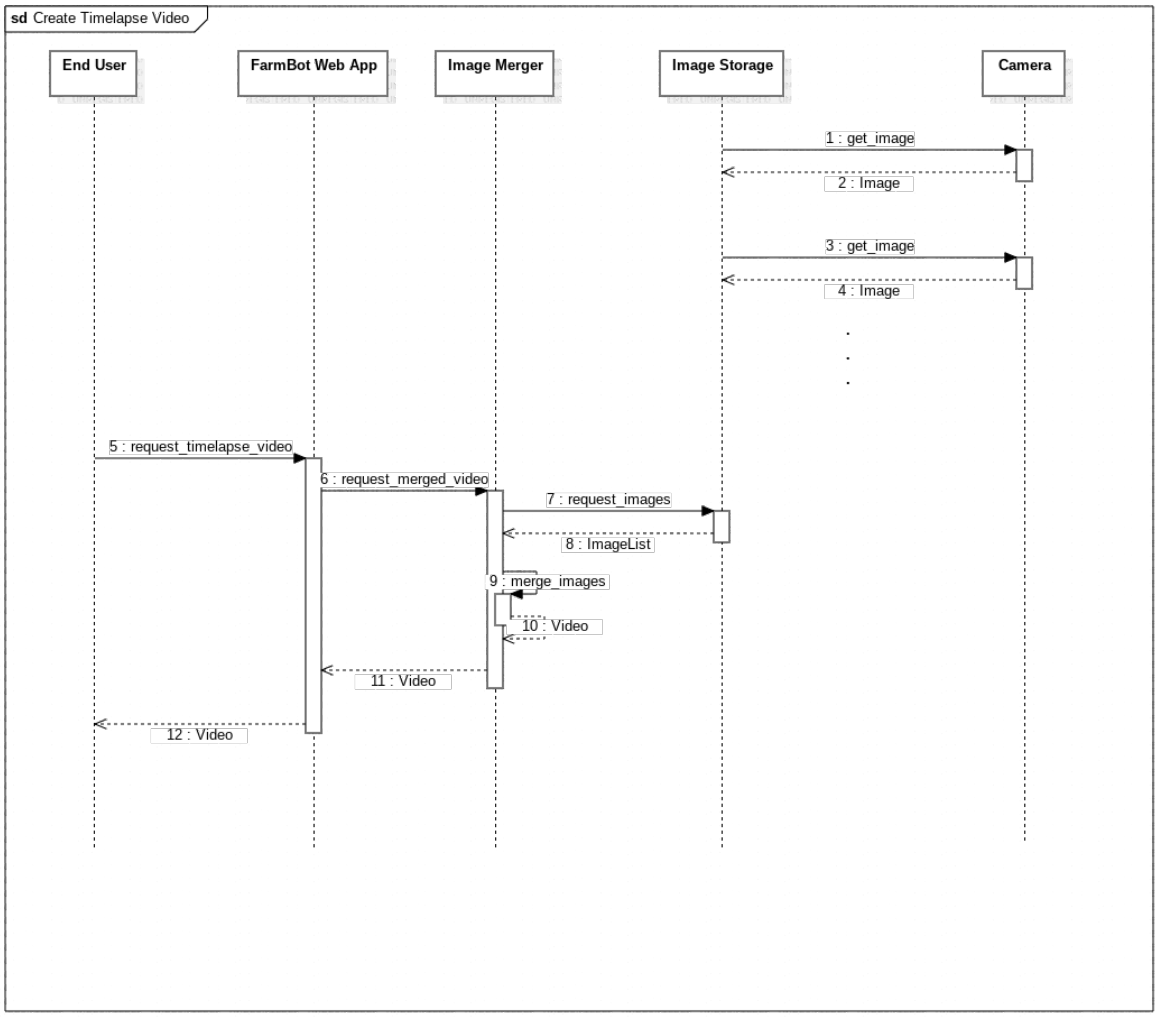

Figure 12: Sequence Diagram for Use Case **Create Timelapse Video**

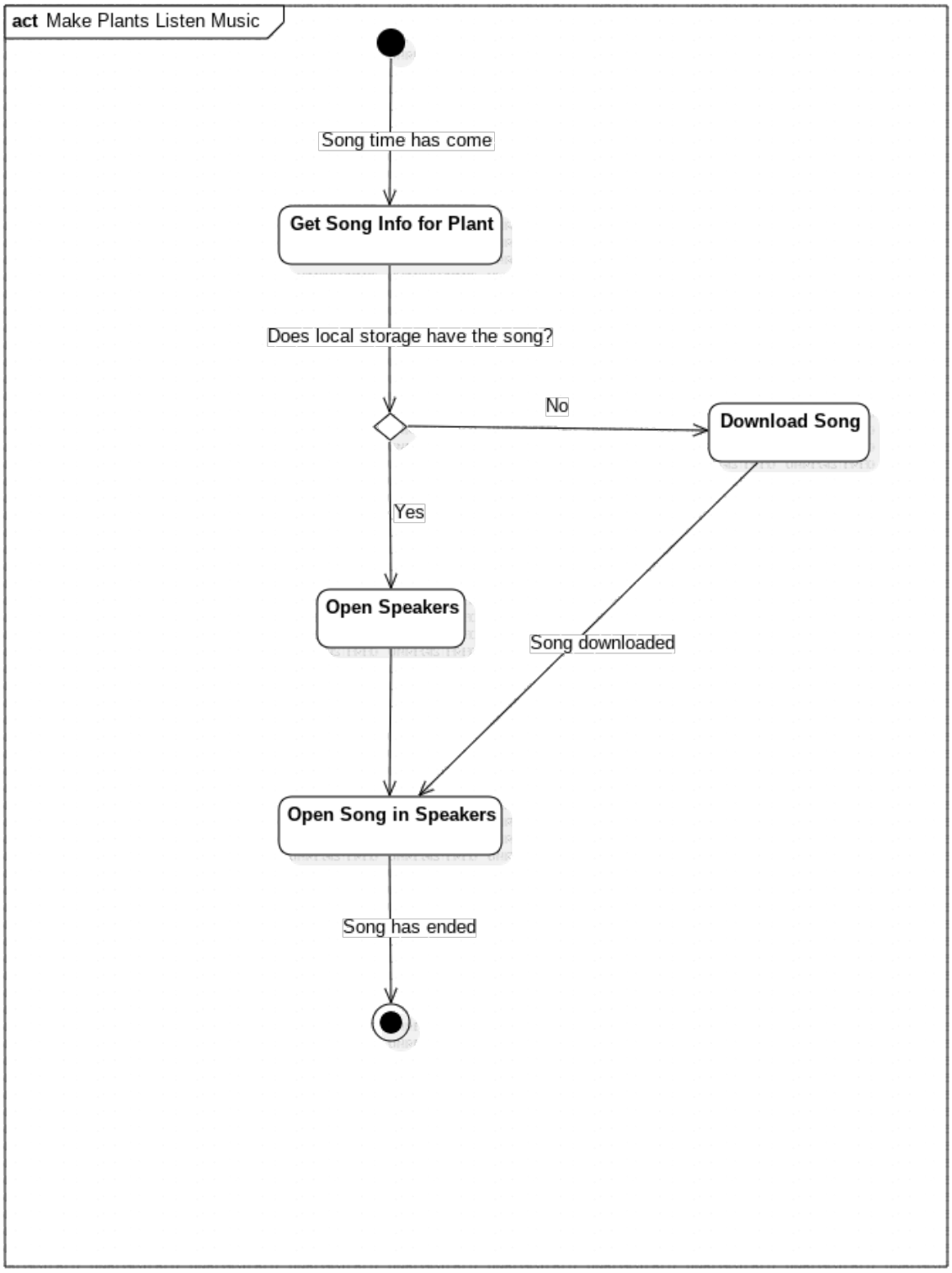

Figure 13: Activity Diagram for Use Case **Make Plants Listen Music**

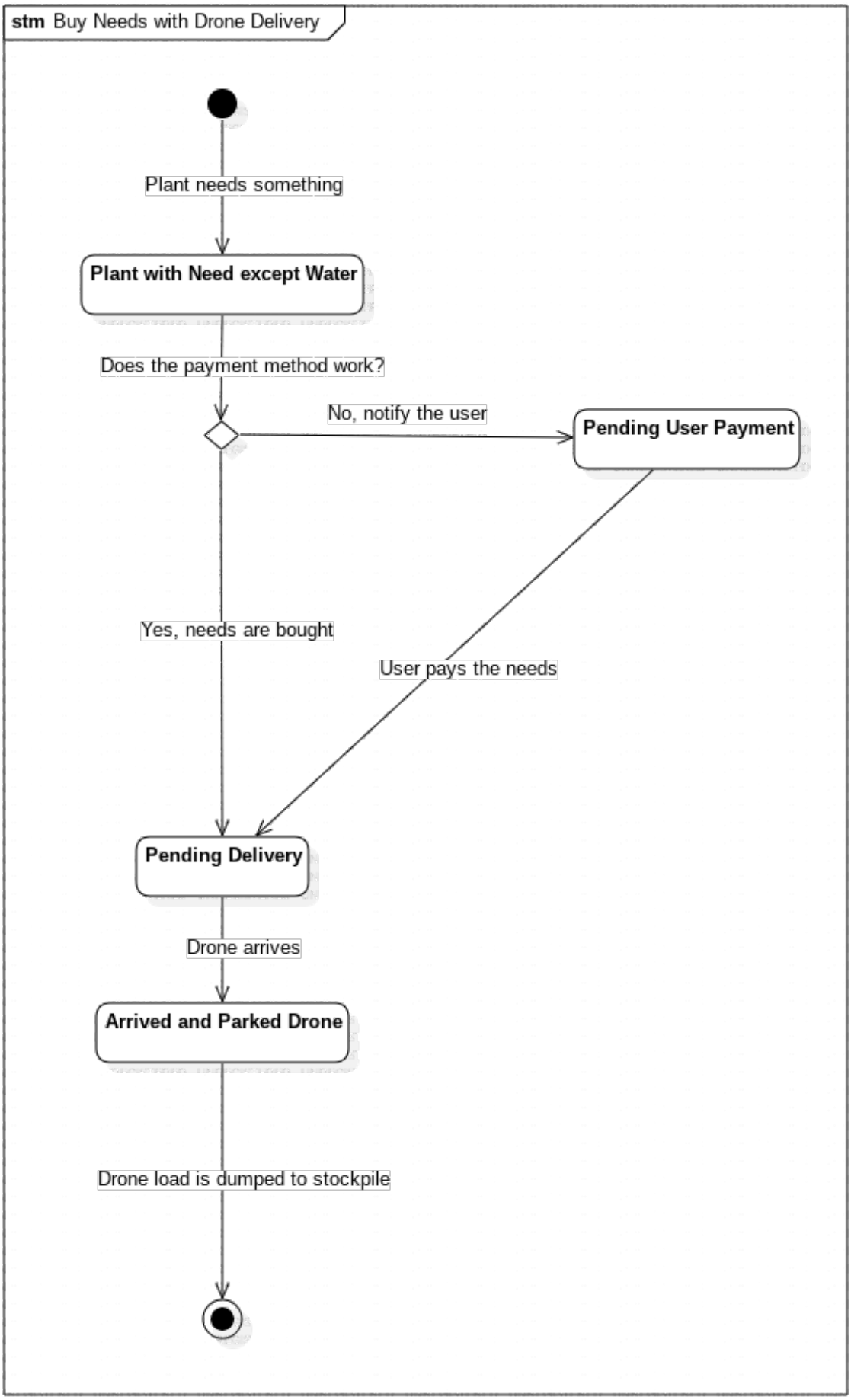

Figure 14: State Diagram for Use Case **Buy Needs with Drone Delivery**

Use Case Name	**Create Timelapse Video**
Actors	Registered User, Sensors and Cameras
Description	This feature outlines the process for the machinery to automatically take imagery of the plants in long intervals and combine them to create a video when the user requests a timelapse video of the plants.
Pre-conditions	User must have access to a web browser and an internet connection. At least one camera must be installed
Data	Name of the camera feed. Time interval of the timelapse video.
Stimulus	User wants to have a timelapse of their garden through camera feeds.
Response	System merges taken images to a video and displays the result to the user.
Normal Flow	1. User accesses the Timelapse Video tab. 2. User provides the camera URL/IP and desired time range. 3. System creates the video from the images by merging them in the background. 4. The resulting video is shown to the user in the Web App.
Alternative Flow	-
Exceptional Flow	Provided camera feed URL/IP is not valid.
Post-conditions	User has a timelapse video of the desired camera feed for the desired time range.
Comments	Ensure that the time range selection is valid that it is inside the earliest and latest available images for the selected camera feed.

Table 12: Tabular description of the **Create Timelapse Video** use case

Use Case Name: Buy Needs with Drone Delivery	
Actors	Registered User, Payment System, Drone Fleet
Description	This feature allows a registered user to purchase items online and have them delivered to their location via a drone.
Pre-conditions	User must have an account on the platform. User's delivery location must be within the drone service area.
Data	User account details, Payment information, Delivery address, Item details.
Stimulus	User selects items for purchase and requests drone delivery.
Response	System processes payment, schedules drone delivery, and provides user with a delivery tracking system.
Normal Flow	1. User logs into their account. 2. User selects items to purchase. 3. User chooses drone delivery option. 4. User enters delivery address. 5. User confirms and pays for the order. 6. System schedules the drone for delivery. 7. User receives items via drone.
Alternative Flow	If the drone delivery is not available, provide alternative delivery methods for the user to choose from.
Exceptional Flow	If payment fails, the order is not processed. If the drone cannot deliver due to weather or technical issues, reschedule delivery or refund the user.
Post-conditions	User receives the items they purchased via drone delivery.
Comments	Ensure the system verifies delivery locations are within drone-operational areas. Safety checks for drone operations should be in place.

Table 13: Tabular description of the **Buy Needs with Drone Delivery** use case

Use Case Name: Store Water from Rain	
Actors	Farmbot Device, Weather Monitoring System, User
Description	This feature enables the Farmbot to autonomously collect and store rainwater for irrigation purposes.
Pre-conditions	Farmbot is operational and has sufficient storage capacity. The Weather Monitoring System is functional.
Data	Weather forecast data, Farmbot's current water storage levels, Farmbot's location coordinates.
Stimulus	Forecast data indicates the likelihood of rain.
Response	Farmbot positions itself to collect rainwater, monitors the collection process, and stores the water in its reservoir.
Normal Flow	1. Weather system predicts rain. 2. Farmbot receives a signal to initiate water collection mode. 3. Farmbot positions collecting apparatus. 4. Rainwater is collected. 5. Farmbot stores the water. 6. User is notified of the collected amount.
Alternative Flow	If the water storage is full, Farmbot can redirect excess water to a designated overflow area or distribute it to areas in need.
Exceptional Flow	If the Farmbot fails to collect water due to mechanical failure or misalignment, an alert is sent to the user for manual intervention.
Post-conditions	Rainwater is collected and stored, ready for use in irrigation.
Comments	The system should regularly check the weather forecast for rain and ensure the Farmbot's water storage capacity is not exceeded. Maintenance checks should be scheduled to ensure the water collection mechanism is functional.

Table 14: Tabular description of the **Farmbot Device Store Water from Rain** use case

Use Case Name: Make Plants Listen to Music [4]	
Actors	Farmbot, User, Audio Playback Device
Description	This feature allows the user to play music for plants as part of a plant care routine to promote growth and health.
Pre-conditions	Audio playback device is installed and functioning. The Farmbot is configured with the user's music preferences.
Data	User's music playlist, schedule for music playback, plant species information.
Stimulus	User selects the option to play music for the plants through the Farmbot.
Response	The system plays the selected music at the scheduled times or upon user's request, with volume adjusted to levels suitable for plants.
Normal Flow	1. User sets up a playlist in the Farmbot. 2. User schedules music playback times. 3. System activates the audio device to play music at the scheduled time. 4. Music is played for the duration set by the user.
Alternative Flow	If the audio device is not functioning, the system notifies the user to check the device or select an alternative playback method.
Exceptional Flow	If there is a power outage or system failure, the scheduled music playback is missed, and the user is notified of the failure.
Post-conditions	Plants are exposed to music according to the user's settings, potentially contributing to a beneficial environment for plant growth.
Comments	Volume levels should be carefully controlled to be gentle and not harmful to plants. Consider the type of music that might be beneficial to plants based on recent scientific studies.

Table 15: Tabular description of the **Make Plants Listen to Music** use case

4.4 Logical Database Requirements

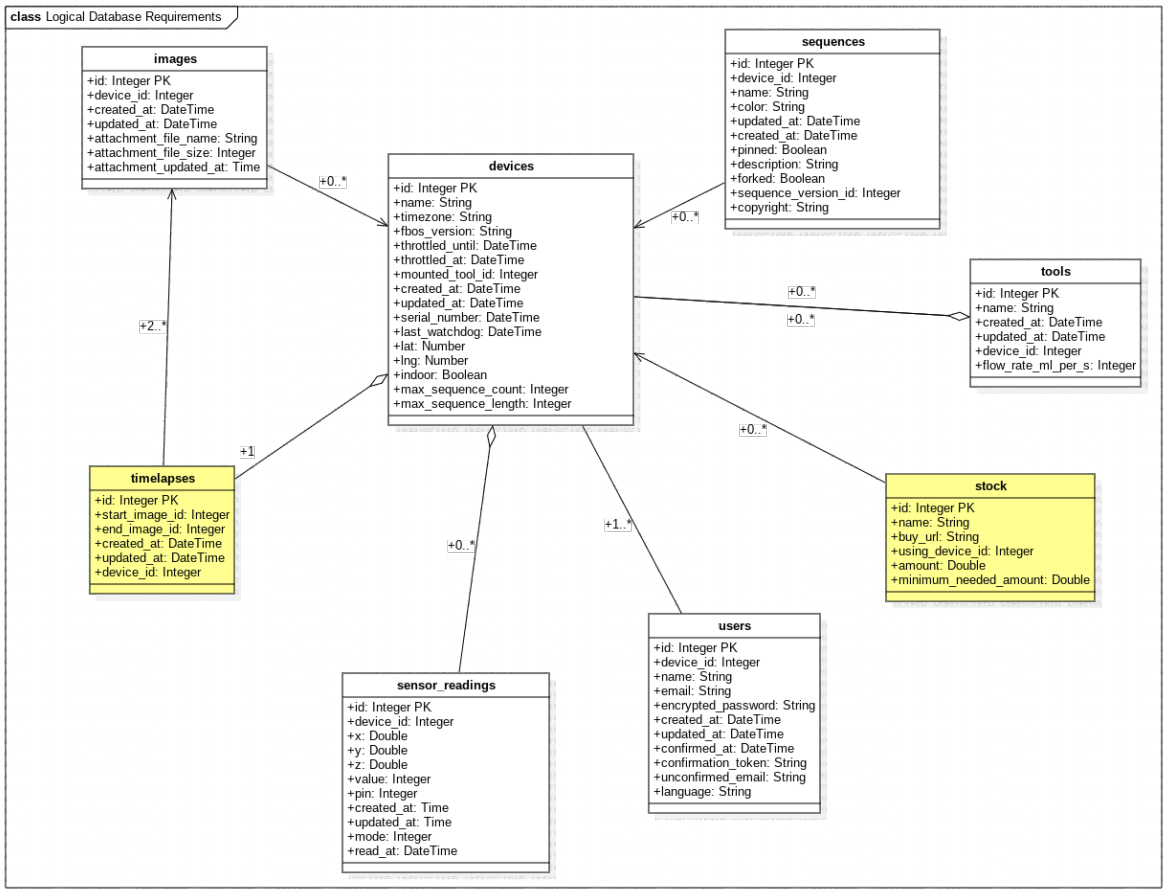

Figure 15: Logical Database Requirements Diagram for Improved FarmBot

Since we included the timelapse feature and Drone delivery to the Farmbot feature. The yellow components, namely the "timelapses" and "stock" are added to the Logical Database Requirements Diagram for Improved FarmBot

4.5 Design Constraints

Beyond the design constraints outlined in section 3.6, additional constraints could also be included:

- **Create Timelapse Video:** Must provide options for user to set image capture frequency and video frame rate.
- **Buy Needs with Drone Delivery:** Required to follow safety and aviation regulatory standards.
- **Store Water from Rain:** Should have sensors to detect rain and initiate water collection systems.
- **Make Plants Listen Music:** Needs to regulate audio levels to maintain plant health safety limits.

4.6 System Attributes

- **Consistency:** Executes regular tasks such as irrigation, seeding, and soil health assessment with minimal interruptions.
- **Operational Readiness:** Accessible for use at any time, characterized by minimal operational pauses and swift error resolution.
- **Data Protection:** Ensures the safety of information and operational controls against unauthorized modifications or access.
- **Efficiency:** Conducts agricultural operations promptly and efficiently, optimizing the deployment of resources.
- **Flexibility:** Can easily accommodate changes or upgrades in technology and farming practices without significant overhauls.
- **Resilience:** Possesses the ability to withstand and quickly recover from natural adversities or technical malfunctions.
- **Customizability:** Offers a range of options that can be tailored to meet the specific needs and preferences of different farming operations.

4.7 Supporting Information

Following our recent suggestions, we're set to significantly upgrade our system, ushering in a new era of efficiency and innovation.

The introduction of a timelapse video feature will allow users to monitor plant growth visually, offering insights into their care strategies and enhancing engagement. Drone delivery for essential supplies promises unmatched convenience, ensuring timely access to gardening needs with minimal effort.

Adopting rainwater storage showcases our commitment to sustainability, providing an eco-friendly water source that's both efficient and cost-effective. Moreover, the pioneering idea of playing music to plants taps into the potential benefits of sonic stimulation on plant health and growth, offering a unique approach to nurturing our gardens.

These enhancements not only streamline operations but also enrich the user experience, marking a significant leap forward in how we interact with and care for our gardening ecosystems. By implementing these suggestions, we're not just improving; we're redefining our connection with nature through advanced technology, leading to a more sustainable, efficient, and engaging gardening future.