Department of Computer Engineering

CENG350 Software Engineering

Software Requirements Specification for FarmBot

Group #61

By

Aytaç Sekmen 2575983

Kerem Karabacak 2644417

Saturday 13th April, 2024

Contents

List of Figures

List of Tables

Revision History

Date	**Reason For Changes**	**Version**
29.03.2024	Initial Setup	0.1
05.04.2024	Final Version	1.0

Table 1: Revision History

1. Introduction

FarmBot is an open source precision agriculture CNC farming project consisting of a Cartesian coordinate robot farming machine, software and documentation including a farming data repository. This introduction part, includes the brief explanation of the FarmBot project and its subsystems.

1.1 Purpose of the System

- The vision of this project is to "Create an open and accessible technology aiding everyone to grow food and to grow food for everyone." [1]
- And the mission of this project is to grow a community that produces free and open source hardware plans, software, data, and documentation enabling everyone to build and operate a farming machine.
- FarmBot attempts to combine the efficiencies of both Monocrop and Polycrop systems into one.
- FarmBot aims to incorporate the machine efficiency, scalability, and the minimal use of labor that traditional equipment takes advantage of while managing a polycrop with great biological efficiency.

1.2 Scope

- The system's front-end will be the primary way the end-user interacts with hardware.

- The user will be able to program the hardware through the system.
- The user will have a opportunity to overview statistics and operations of hardware through the system's dashboard with useful functionalities.
- The system will provide data maps to users with the data coming from the hardware for smarter decisions.
- The system will use OpenFarm data, which is a database for farming and gardening knowledge, to decide. [2]
- The user will be able to control the hardware manually in case of any troubleshooting need through the software.
- The user will need to create a account and link this account with a hardware to use the hardware's functionalities.
- The system will provide data storage service to user (size, statistics, data maps, history etc.) in databases, and this data will be used to provide a decision support system (seed spacing, plant pairing, watering etc.).
- The system will store the equipment's profile(configuration) in the database in case of any reliability issues.
- The system microcontroller will provide communication between backend and hardware with its own operating system that can interpret the numerical code that comes from backend.
- The system will provide an open data sharing subsystem to help other farmers in decision-making, and it will be reachable from front-end of the system.
- The system will not provide any service for product selling or advertising.
- Also, users will not reach each other's profiles or follow someone i.e. any social media-like service is not in the scope of the system.

1.3 System Overview

This part includes the detailed explainations of the subcomponents in the FarmBot project.

1.3.1 System Perspective

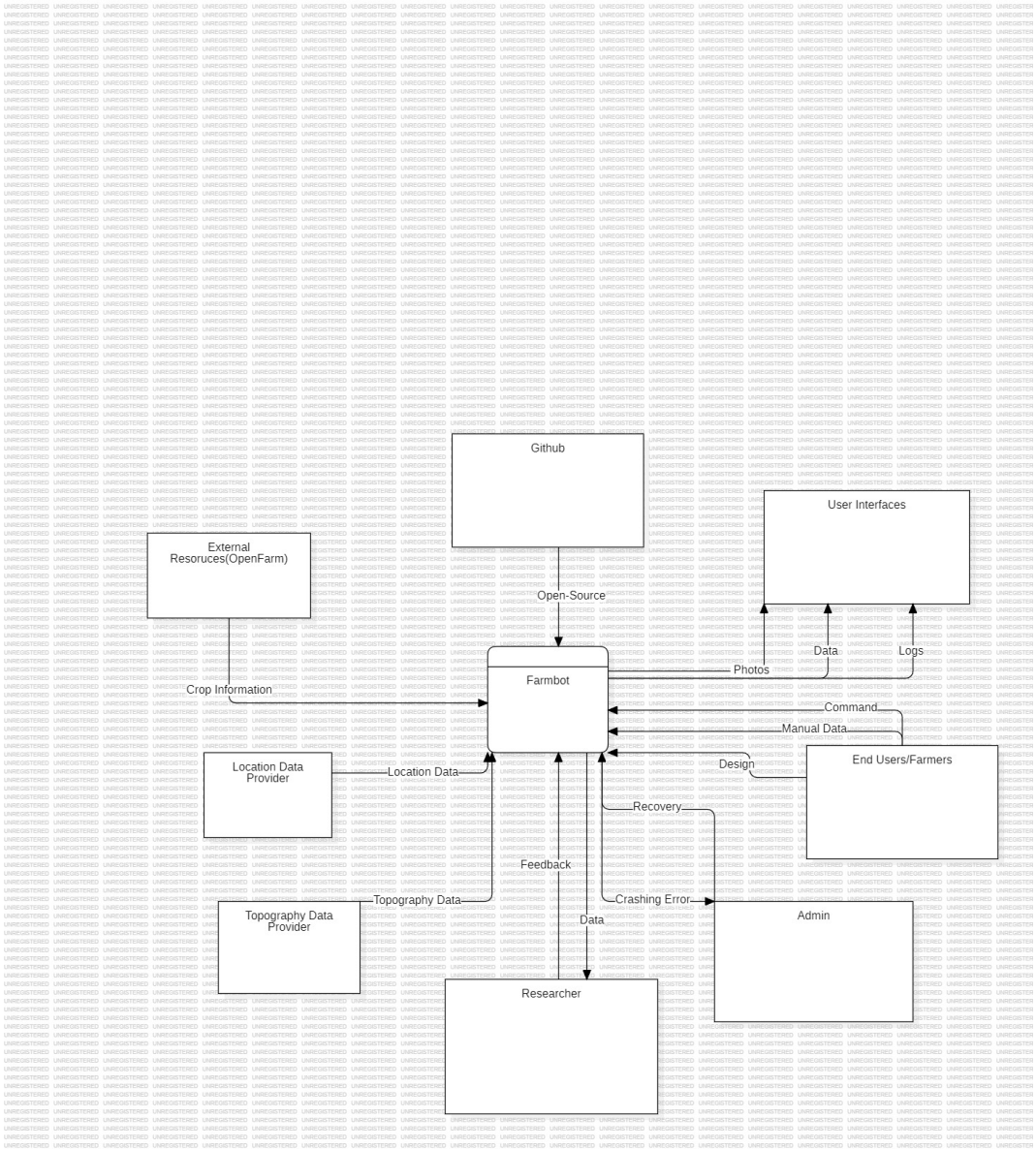

Figure 1.1: Context Diagram

- The context diagram includes all the components that is external to system i.e. these components are not actually in the system, but interacts with the system. For Farmbot point of view, these components are Farmers/Users, User Interfaces, Researcher, Topography Data Provider, Location Data Provider, Github, External Resources(OpenFarm database).

- The users of Farmbot will be able to control the hardware through the system with commands. Also manual data input will be possible for more effective decision system.

- Users will do these operations through the system's interfaces, and these interfaces will include photos, datas and logs coming from the field, hardware.

- Researchers will be able to use the system and the data. Thus, they will be able to provide feedback to developers for better user experience and more reliable decision system.

- The decision support system needs a lot of different kind of information. Some of them will come from OpenFarm, and some of them will come from external sources such as topography data and location data.

- Since Farmbot is a open-source project, all the source code will be stored in Github, which is the most popular site for this purpose.

- Farmbot developers have also created a project called OpenFarm, which contains all the useful data for the decision-making system. This project was specified as an external project to Farmbot. It's also open-source and can be found in Github.

- Admins will recover crashes and have special type of profiles. They can fix this errors thorugh admin dashboards and interfaces. Also they will update current state of Web App.

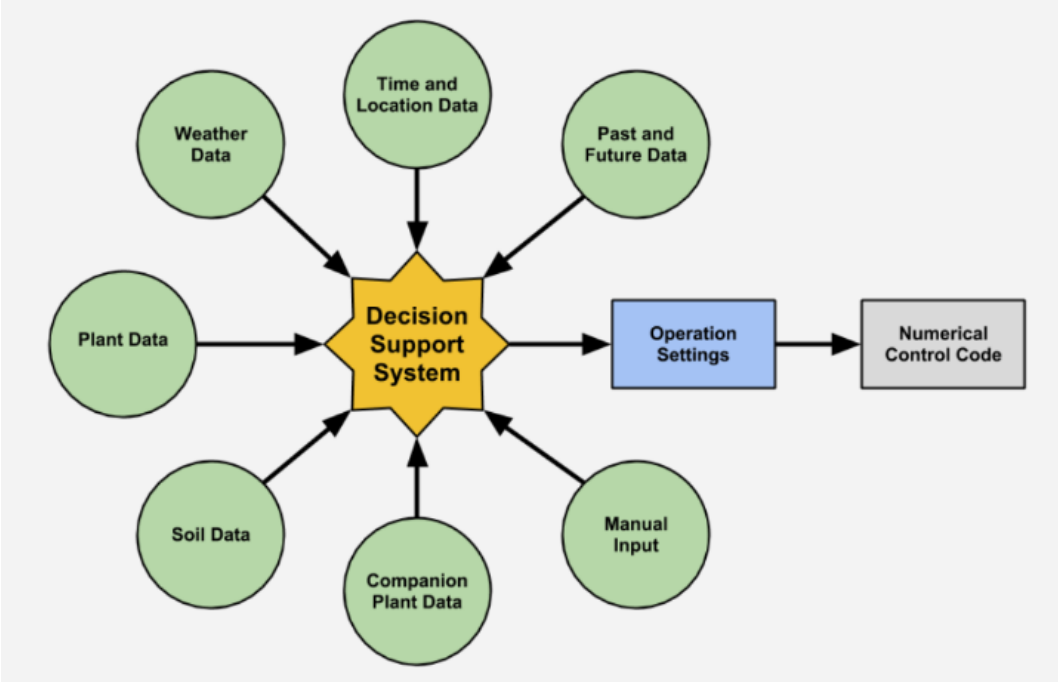

Figure 1.2: Decision Support System

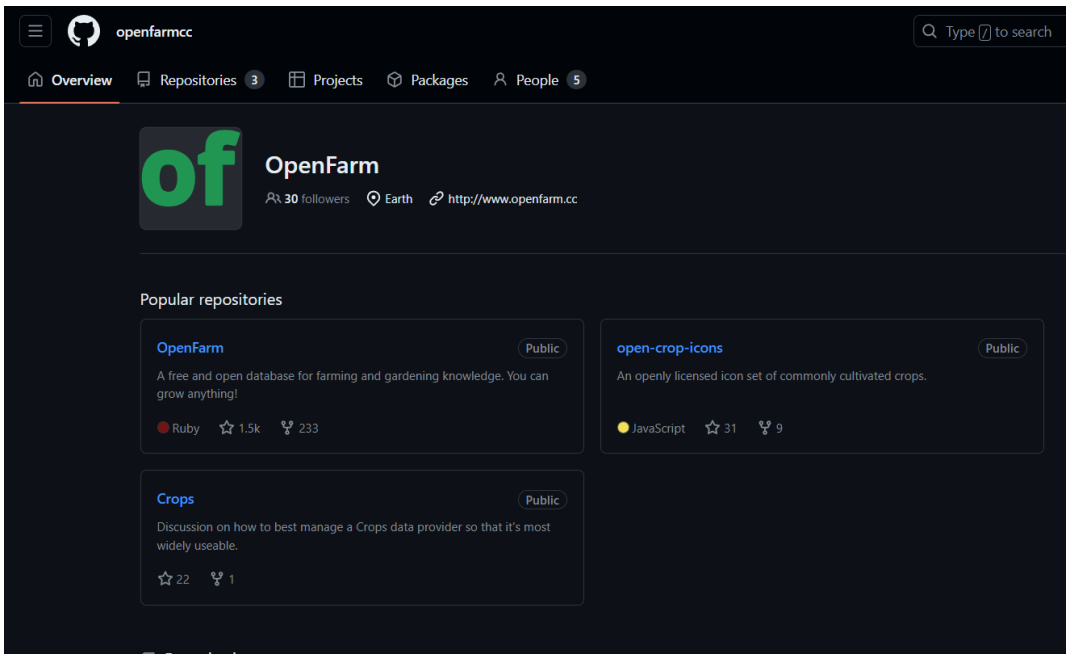

Figure 1.3: OpenFarm Github Page

1.3.1.1 System Interfaces

The list of interfaces will be given here, but details will be given in further sections.

- Plants Tab (UI)
- Weeds Panel (UI)
- Points Panel (UI)
- Curves Panel (UI)
- Sequences Panel (UI)
- Regimen Panel (UI)
- Events Panel (UI)
- Data Usage Interface (HI)
- Configuration Interface (HI)
- Connectivity Panels (CI)
- OS Update Panel (SI)
- MQTT Gateway (SI)
- Github (SI)
- Control Panel (CI)
- Raspberry Pi Imager (CI)
- Sensors Panels (CI)
- Photos Panel (CI)
- Tools Panel (CI)

1.3.1.2 User Interfaces

The Farmbot user interfaces will be modern, simple and powerful to use. Each interface will be functional as much as possible. With that user won't lost between a lot of different tabs and will be able to do whatever he/she wants quickly.

- The user can view all of his/her plants in the plants panel, as well as in the map. The user also can add plants, drag and drop the crop image into the map to create a plant in garden, and delete.

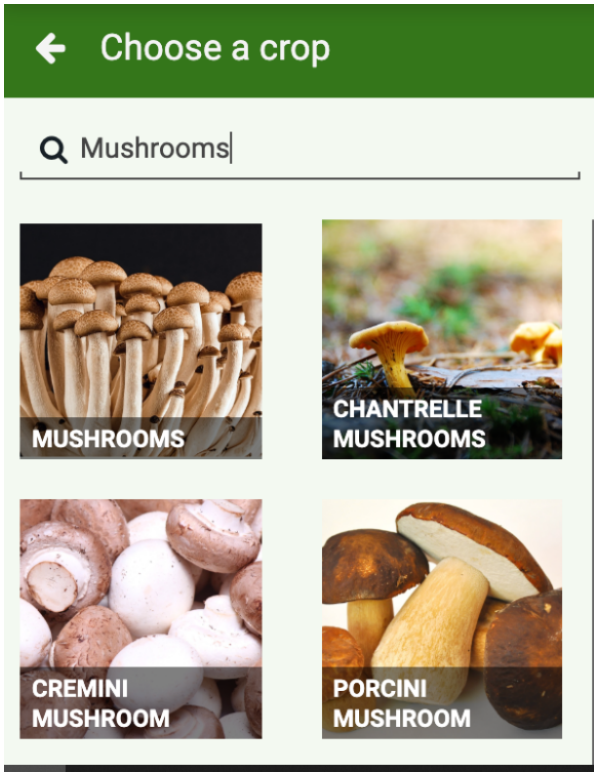

Figure 1.4: Crop Choosing Interface

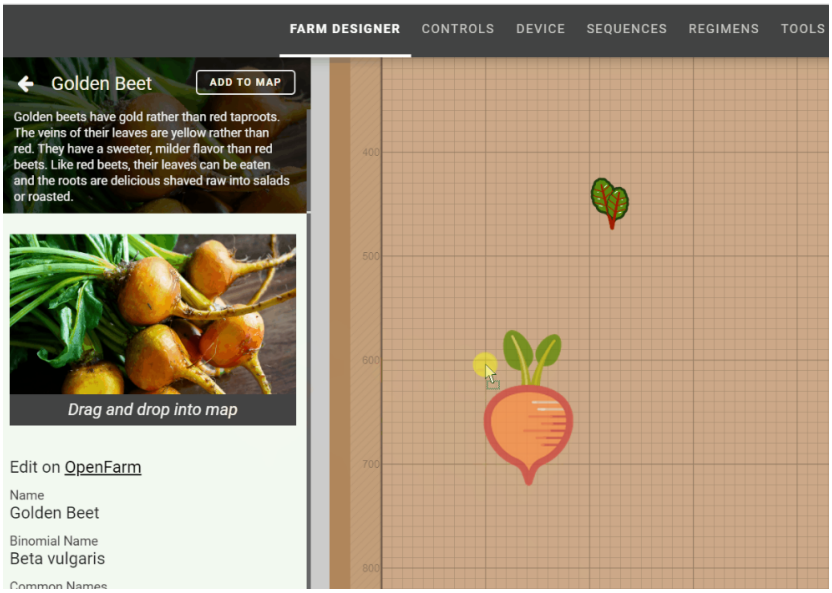

Figure 1.5: Drag and Drop Interface

- The weed panel will allow the user to manage the weeds in his/her garden by adding, editing etc.

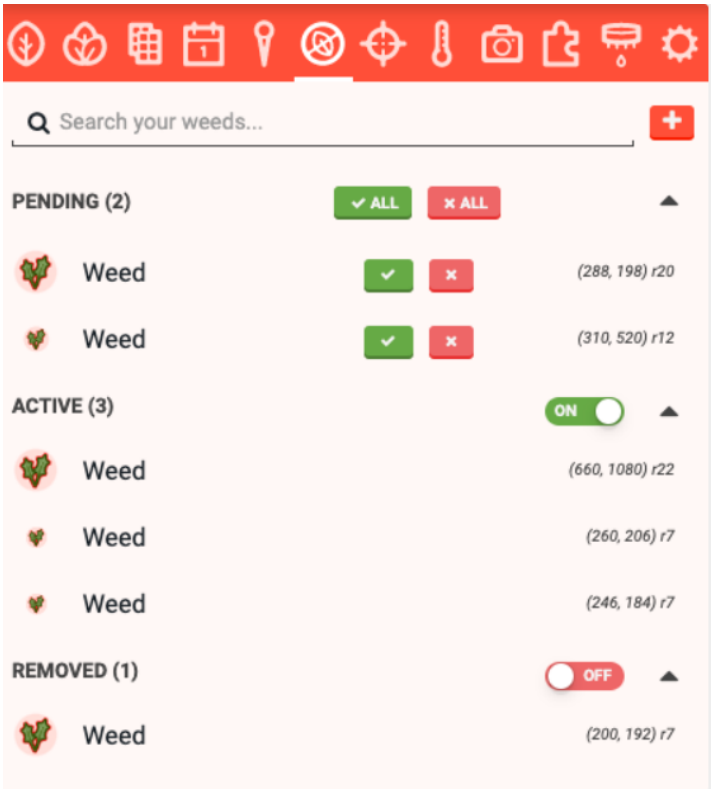

Figure 1.6: Weeds Panel Interface

-The points panel will allow the user to create custom locations for Farmbot to travel non-plant destinations.

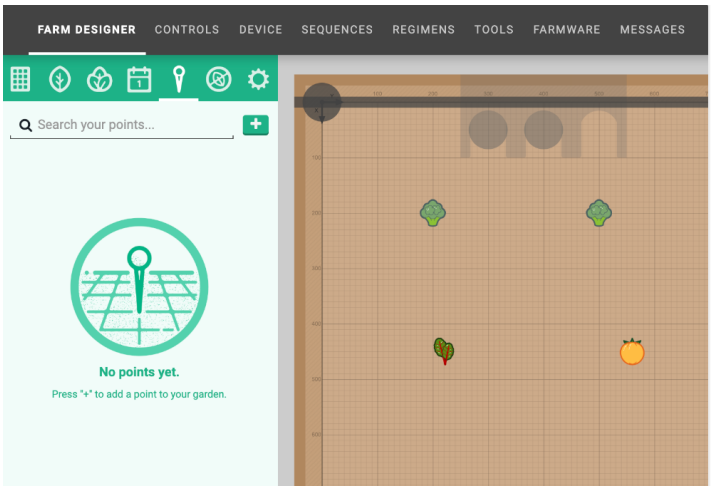

Figure 1.7: Points Panel Interface

- The curves panel will allow the user to view plots of data that shows how a need changed such as water, spread, height.

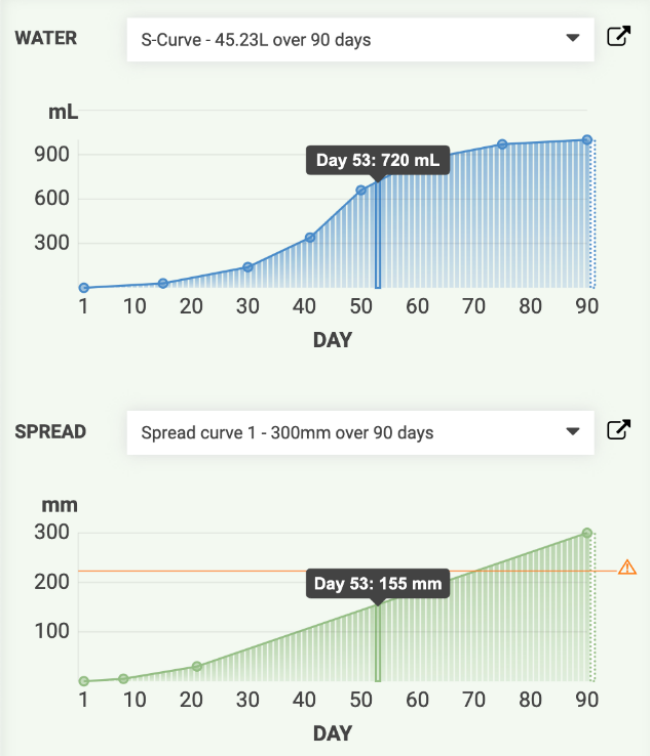

Figure 1.8: Curves Panel Interface

- The sequences panel will allow the user to combine the commands to create more complex actions. The Farmbot will execute this commands sequentially until the end.

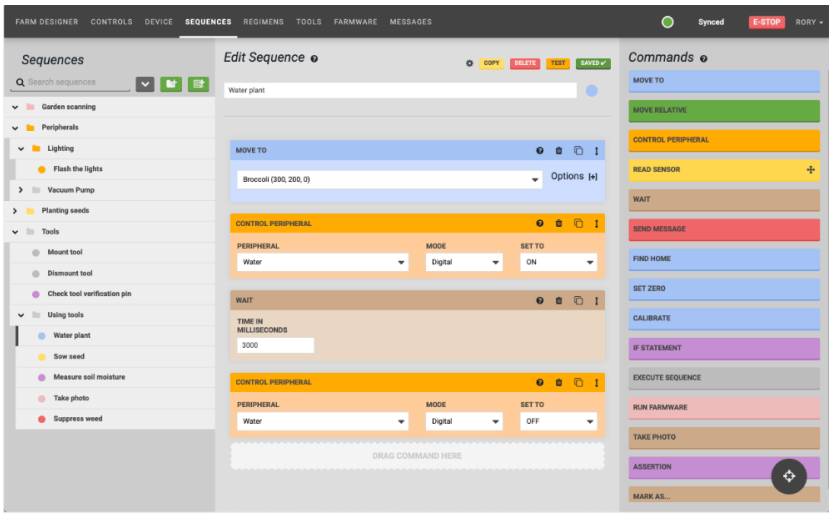

Figure 1.9: Sequences Panel Interface

- The regimens panel will allow the user to create a recipe to use season after season. More general compare to events.

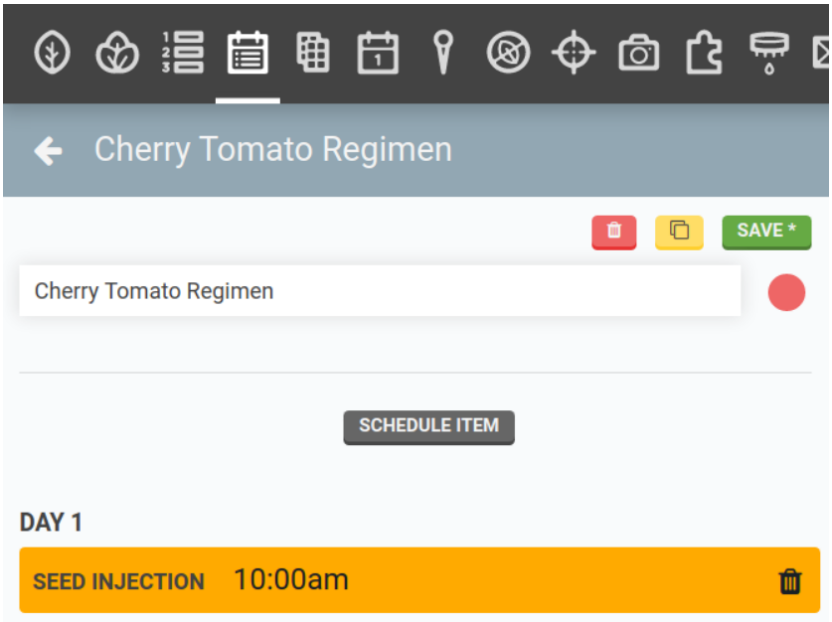

Figure 1.10: Regimen Panel Interface

- The events panel will allow the user to execute any sequence or regimen automatically by setting it, and reuse.

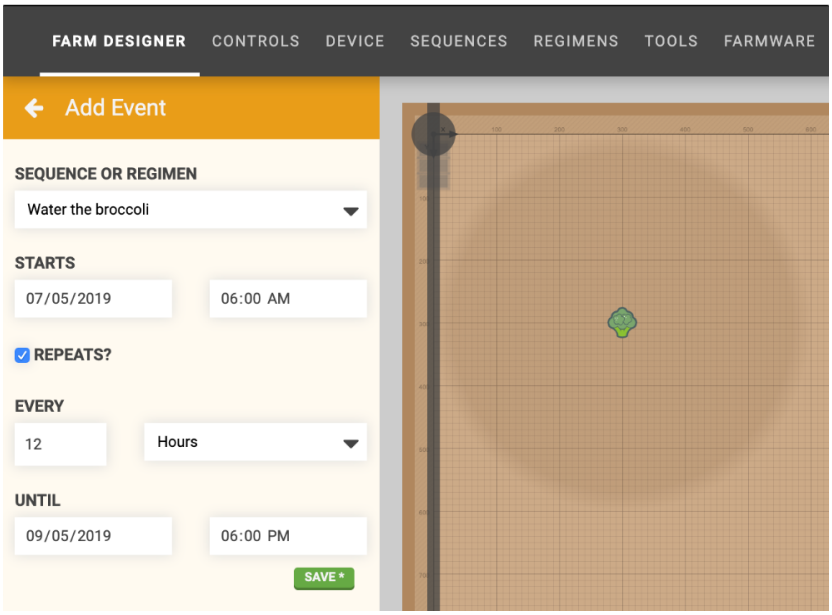

Figure 1.11: Events Panel Interface

1.3.1.3 Hardware Interfaces

- Configurator is a piece of software built into FarmBot OS that makes it easy to connect user's FarmBot to a WiFi network and user's web app account.

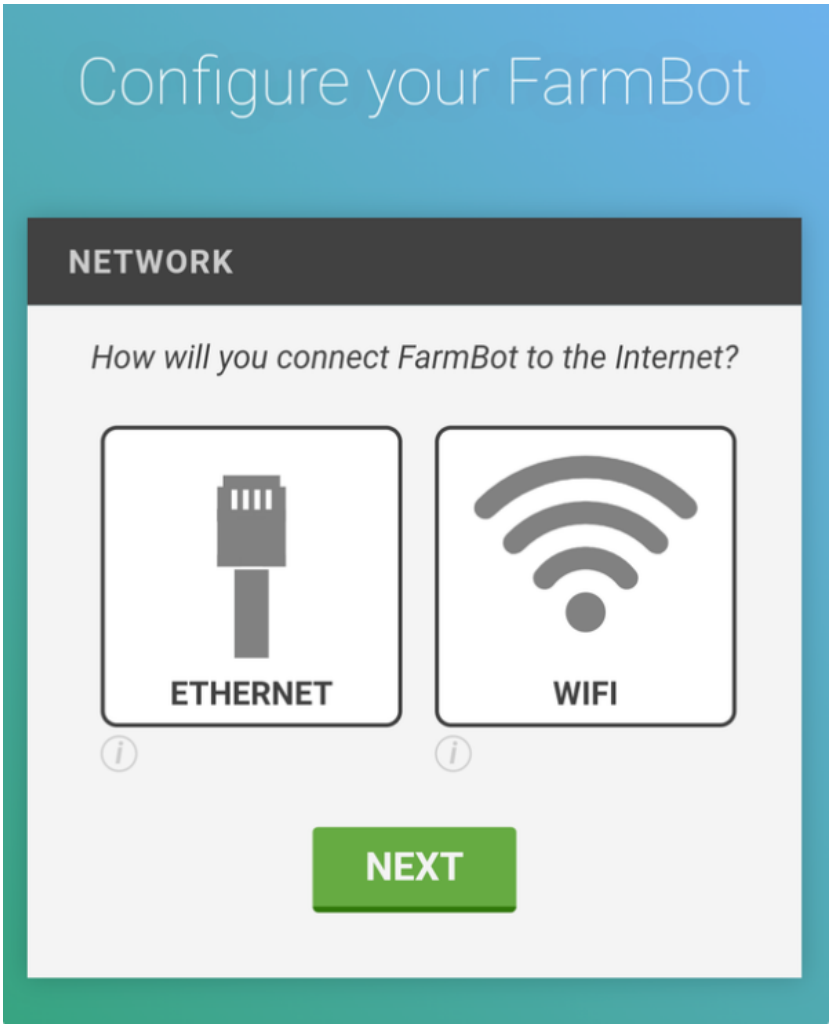

Figure 1.12: Configuration Interface

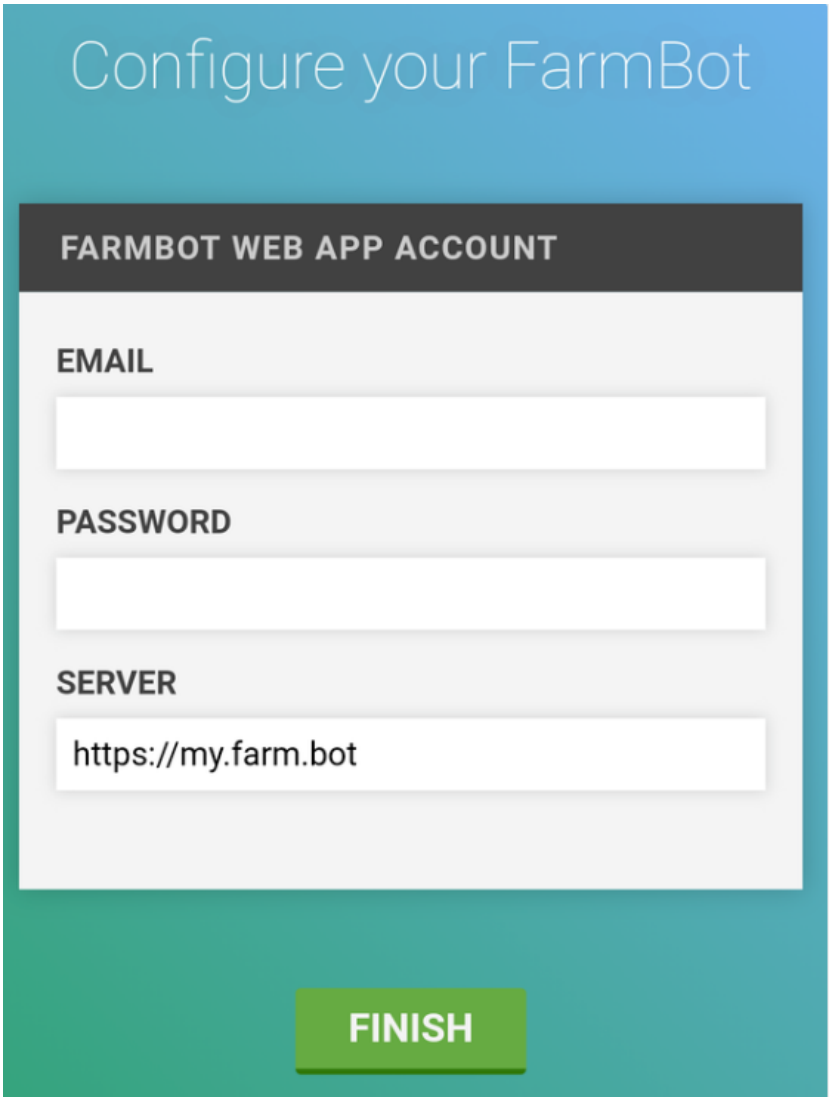

Figure 1.13: Configuration Interface

- The data usage panel will show the average amounts of data transmitted and received by the FarmBot to complete various tasks.

- The interface of instruction set of FarmBot Operating System. Some instructions can be changed from here manually.

1.3.1.4 Software Interfaces

- The user will be aware of upcoming updates, and set update time. Also the user will reach release notes with a pop-up.

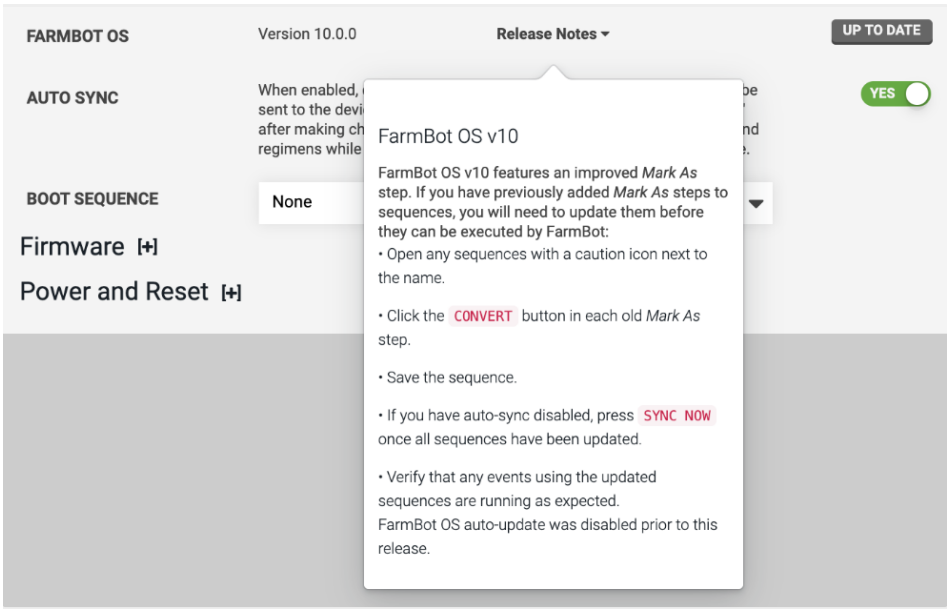

Figure 1.14: Update Panel Interface

-The MQTT Gateway is a cloud application that acts as an intermediary for all messages between the FarmBot web app and FarmBot devices. It handles socket connections, device identification, and authentication.

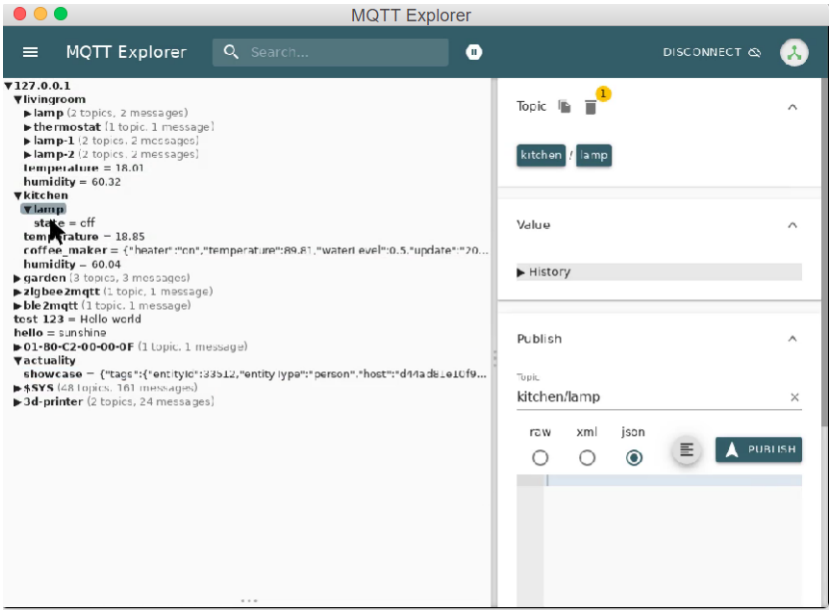

Figure 1.15: MQTT Gateway Interface

- The interface of FarmBot Operating System. Some settings can be changed manually.

- Since Farmbot is an open-source project, all the codes are available on Github page. [3]

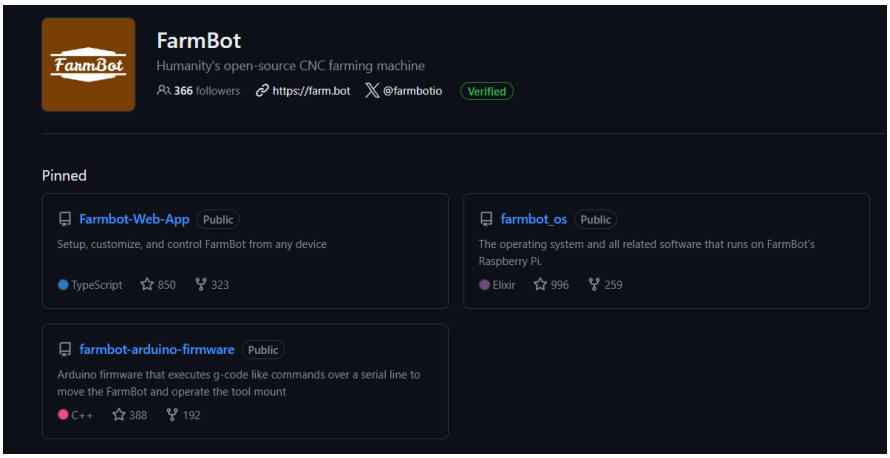

Figure 1.16: Github Interface

1.3.1.5 Communication Interfaces

- The realtime tab will show information about user's computer's and FarmBot's current connection to the web app and message broker, as well as some vital health metrics such as the CPU temperature and voltage of the Raspberry Pi. The network tab will show Wifi signal and IP address, and lastly history tab will show health data of hardware from the past 24 hours.

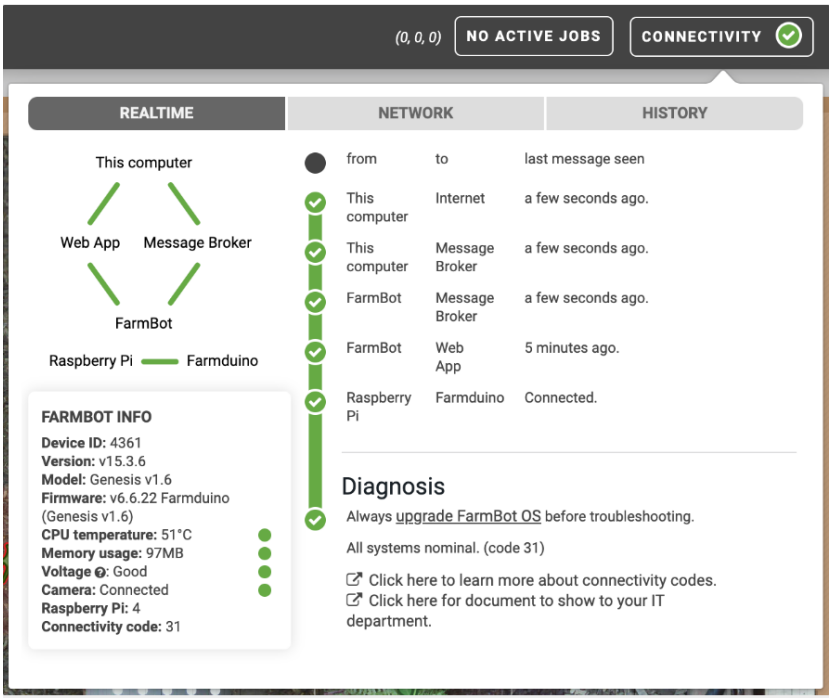

Figure 1.17: Realtime Panel

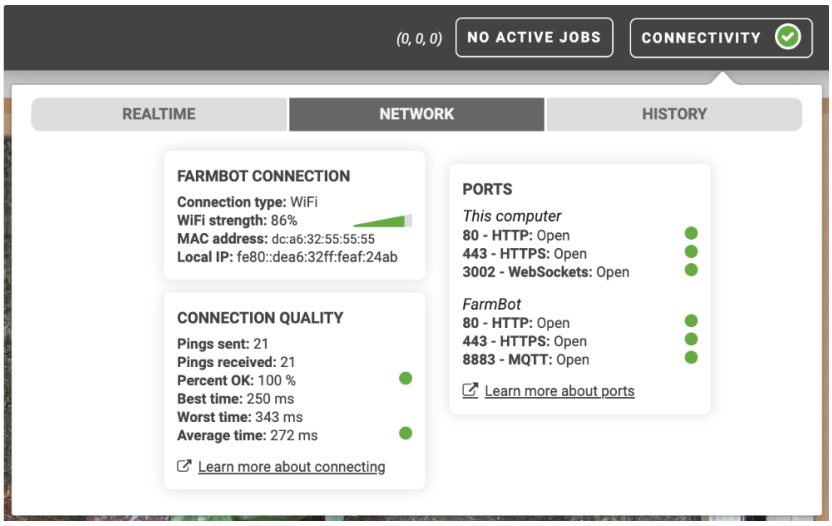

Figure 1.18: Network Status Panel

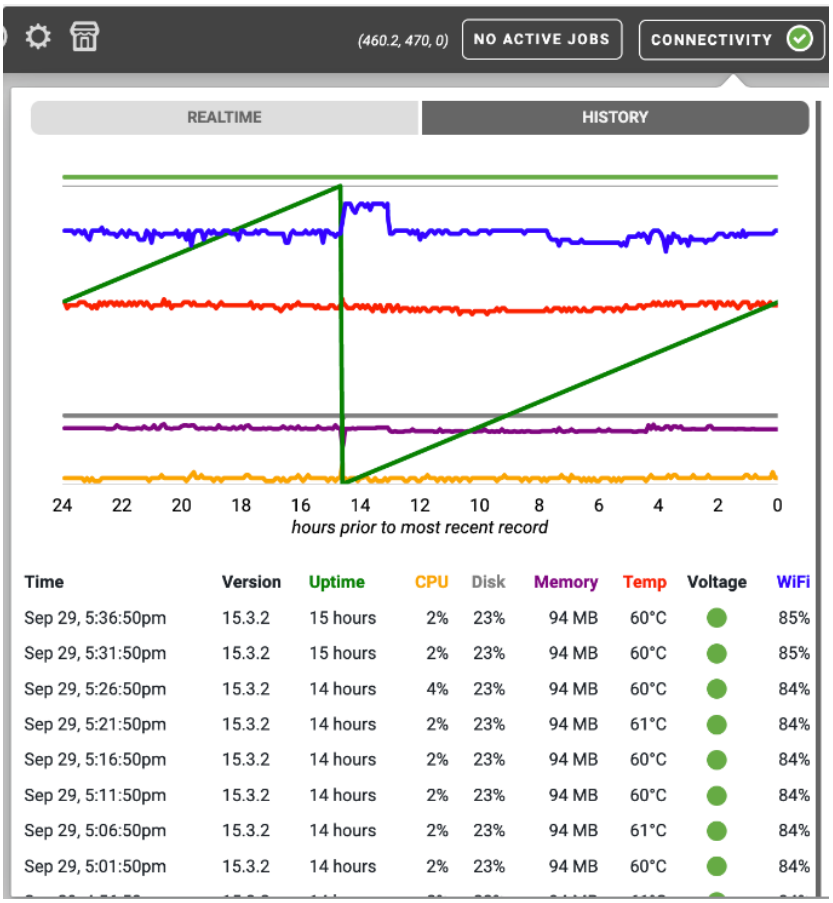

Figure 1.19: History Panel

- Raspberry Pi Imager is essential to write FarmBot OS onto the microSD card. The user will need to connect the microSD card to his/her computer using a card reader to write FarmBot OS.

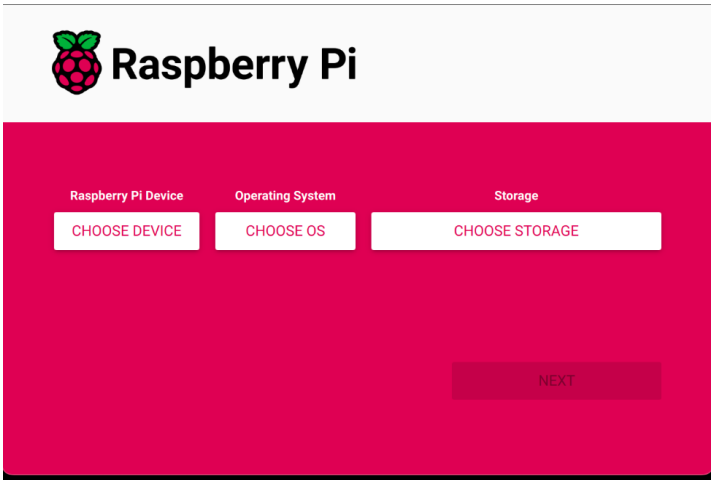

Figure 1.20: Raspberry Pi Imager Interface

- With control panel, user will be able to control the hardware manually. Functionality of this panel is moving the hardware, livestream through the webcam etc.

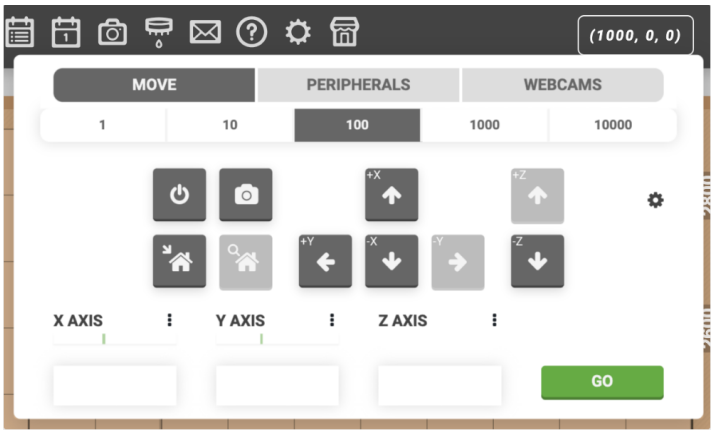

Figure 1.21: Control Panel

- Sensor panels will allow to create new sensors, reading the sensors, view historical operations, and deleting the sensors manually to user.

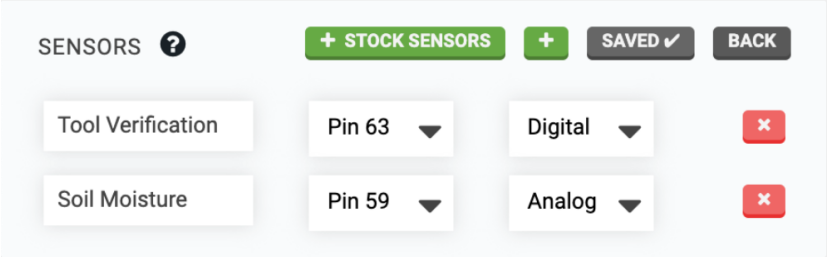

Figure 1.22: Sensor Creation Interface

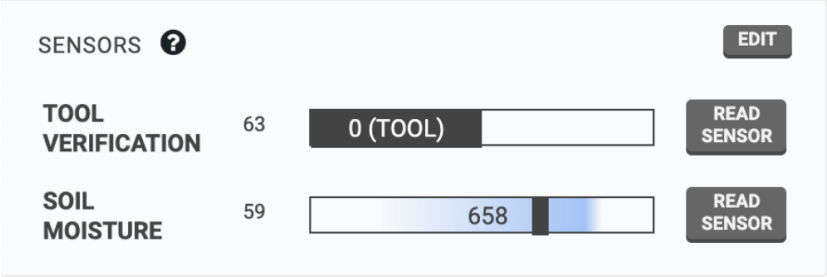

Figure 1.23: Sensor Reading Interface

- In the photos panel, the user will be able to view the photos came from the hardware from oldest to newest. The user will be able to take photos manually from this panel.

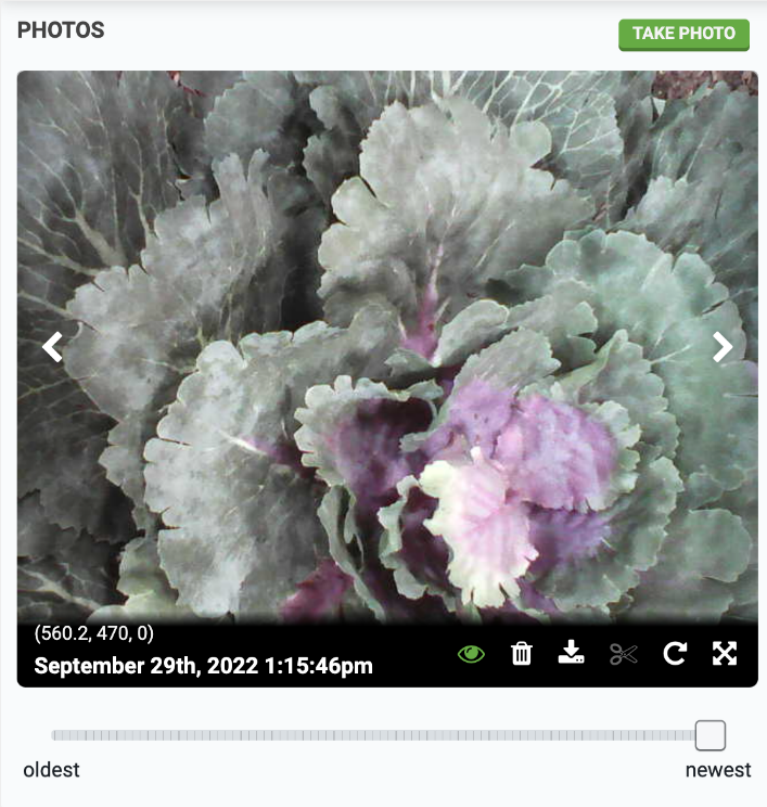

Figure 1.24: Photos Panel Interface

- On the tools panel the user will be able to manage all of his/her FarmBot's slots, tools, and seed containers as well as view the currently mounted tool.

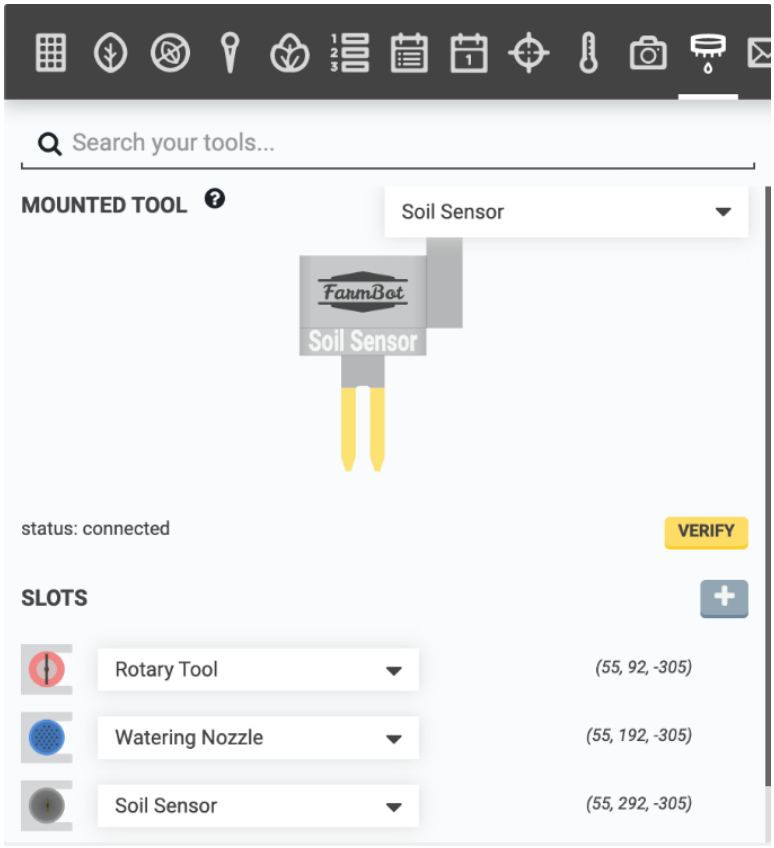

Figure 1.25: Tools Panel Interface

1.3.1.6 Memory Constraints

- Farmbot includes two different devices that have memory. Raspberry Pi and Ardunio Mega 2560.
- Raspberry Pi needs to store photos coming from webcam, and log, sensor data coming from Ardunio Firmware temporarily before sending it to MQTT Gateway. So Raspberry Controller needs to have sufficient memory size for these storage issues.
- Ardunio hardware needs to store data coming from sensors, and rotary encoder data temporarily before sending it to Raspberry Pi Controller. So Ardunio needs to have sufficient memory size for this temporary data storage process.
- Farmbot's cloud services need to have sufficient memory to store all farmer profile data, all user's logs, sensor data and photos.

- Also OpenFarm's database needs to have enough memory to store all crop information.

1.3.1.7 Operations

- Take Photo
- Calibrate Camera
- View Photos
- Add Regimen
- Add Sequence
- Verify Email
- Add Curve
- Install Farmbot OS
- Setup Product
- Add Plant
- Login
- Create Account
- Read Sensor
- Add Sensor
- Add Weed

All these functions' details and use cases are covered in Functions section (3.3).

1.3.2 System Functions

- In this section, major system functions will be covered in summary. The details are available again in Functions section (3.3).

- In order to use FarmBot, every user needs to create an account (Create Account function), and login every time with login credentials which was declared while creating the account (Login function) through user interfaces. Also, the user needs to verify his/her email through user interface. (Verify Email function)

- In order to activate the Farmbot hardware, the user needs to install Farmbot OS

with the help of Raspberry Pi Imager program, (Install Farmbot OS function) and set up the product i.e. complete configuration steps. (Setup Product function)

- After these set-up processes, user will be able to use system functionalities through FarmBot App User Interfaces. (Add Seqeunce, Add Weeds, Add Regimen, View Photos...)
- The user also will be able to communicate with the hardware thorugh user interfaces and some other helper programs such as MQTT Gateway. For example, Take Photo function. The user will be able to take photo i.e. send request to hardware, through interface. Same logic is also available for other functions (Calibrate Camera, Read Sensor...)
- For more details please refer to Functions section (3.3).

1.3.3 Stakeholder Characteristics

- Main stakeholders group is, of course, farmers/end-users. They will spend their time in the user interfaces mostly. The users may not be prone to complex interfaces. So interfaces must be simple and useful as much as possible.
- These interfaces will be developed by developers, which is another stakeholder group. They must be experienced and talented in order to create these simple and useful interfaces. Since the project is open-source developers should not expect high income during developing process and after.
- Another stakeholder group is researchers. The researcher that is giving the feedback must be knowledgeable in his/her research area. For efficient feedback process, researchers should find whatever they are looking for in the Farmbot app. So data interfaces should be useful and easy to use.

1.3.4 Limitations

- Not all data may be collected due to some regularity restrictions. This can affect Decision Support System in a bad way.

- Since the hardware will collect sensor data indefinitely, the memories of Ardunio and Raspberry should be enough to temporarily store all the data. Any overflow may cause data loss and cause some subsystems to shut down.
- Other application interfaces may shut down temporarily or indefinitely. e.g. Location Data Provider. This will lead to some problems in the Decision Support System.
- Concurrent execution of multiple tasks may consume so much resources from the system. This can lead to efficiency problems.
- Any problem in the audit functions such as Logging and tracking of farming activities, including planting schedules can lead to incorrect data and affect the whole system.
- There is no limitation on control functions.
- Some subsystem may not support some high-order languages, and this can affect the system in a bad way.
- Any hardware damage or bug in signal handshake protocols may affect the data flow between hardware devices and the software.
- Any hardware damage arise due to environmental factors may affect the system.
- Any crucial subsystem error,such as MQTT Gateway, which is enabler of data transfer, may affect the whole system since all the data flow will stop.
- In the case of hacking, all the user information, data can be captured by hackers. Since the system is open-source, source code security is not important.
- The interfaces must be easy to use. Any interface design which is hard to use can remove the user from the system or decrease the efficiency of system due to user errors for that specific user.
- Any data latency from other systems that is not present in the project directly such as Topography Data Provider, can lead to misinterpretation in Decision Making System.

1.4 Definitions

UI	User Interface
HI	Hardware Interface
SI	Software Interface
CI	Communication Interface

Table 1.1: Definitions

2. References

[1] R. Aronson, "Farmbot." https://hackaday.io/project/2552-farmbot-open-source-cnc-farming/details, August 2016.

[2] G. Burnworth, "Openfarm." https://github.com/openfarmcc/OpenFarm/wiki, April 2018.

[3] Farmbot, "Farmbot." https://github.com/FarmBot, March 2013.

3. Specific Requirements

3.1 External Interfaces

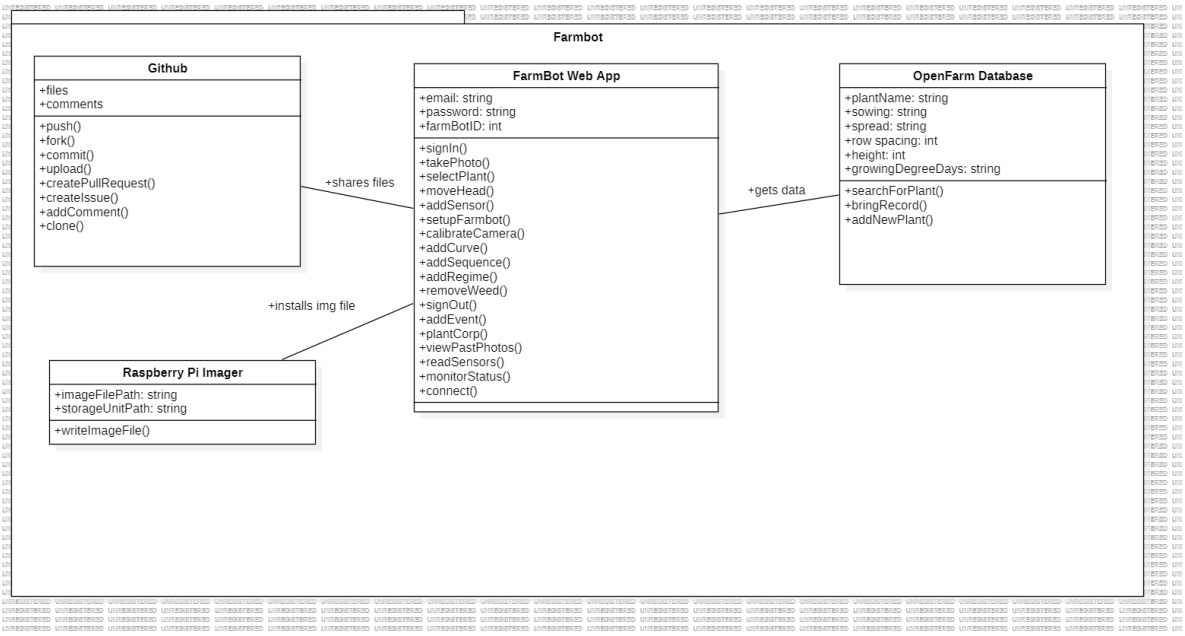

Figure 3.1: External Interfaces

- **Raspberry Pi Imager** is an user-friendly tool developed by Raspberry Pi Foundation for creating bootable media for Raspberry Pi devices. The user will need to connect the microSD card to his/her computer using a card reader to write FarmBot OS into microSD.

- **Github** is a online software development platform. Since Farmbot is a open-source project, all the source code will be stored in Github, which is the most

popular site for this purpose.

- **OpenFarm**, contains all the useful data like plant's features such as estimated spread, height value etc. Even though this project was also developed by Farmbot engineers, it is specified as an external project to Farmbot in their whitepaper. It's also open-source and can be found in Github.

3.2 Functions

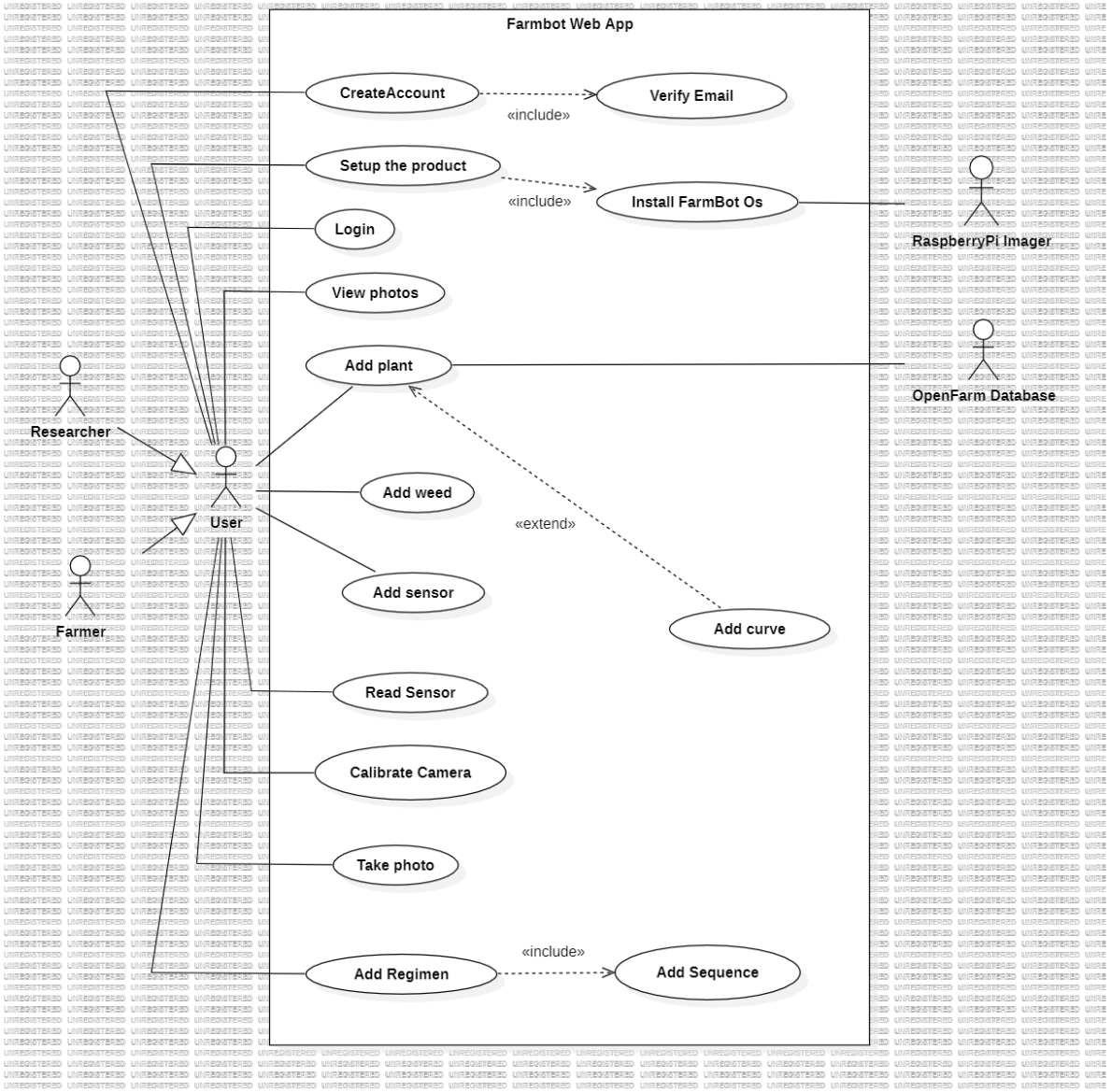

Figure 3.2: Use Case Diagram

Use-Case Name	Take Photo
Actors	User
Description	The user takes photo of the system.

Data	Taken Photo
Preconditions	User must be connected to the system. Camera must be connected to system and be available.
Stimulus	The user clicks on "Take Photo" button
Basic Flow	Step 1 – The user clicks on "Take Photo" button under "Photos" section. Step 2 - The user send request to take picture. Step 3 - The system saves the image when the button was pressed into an image file in SD Card. Step 4 - The user sees the picture.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	The photo which contains the data when researcher pressed the button must be saved.

Table 3.1: Take Photo

Use-Case Name	Calibrate Camera
Actors	User
Description	The user calibrates the camera
Data	Paper including grid of white circles
Preconditions	1. User must be connected to the system. 2. Camera must be available 3. Calibration card must be available
Stimulus	The user clicks on "Read Sensor" button

Basic Flow	Step 1 – User should place the camera calibration card face down on the soil underneath the camera, with the grid of white circles facing up. Step 2 - The user clicks on the "Calibrate" button and send request to calibrate the camera. Step 3 - The system calibrates the camera and process is done.
Alternative Flow#1	Step 1 – If camera calibration card is not available, then user should place two red calibration objects on the surface of the soil in their garden bed. The objects should be bright red, and preferably round. Step 2 - User should enter the distance from one calibration object to the next one and the color range of the red objects manually. Step 3 - User presses the calibrate button. Step 4 - The system calibrates the camera and process is done.
Alternative Flow#2	-
Exception Flow	-
Post Conditions	Camera must be calibrated.

Table 3.2: Calibrate Camera

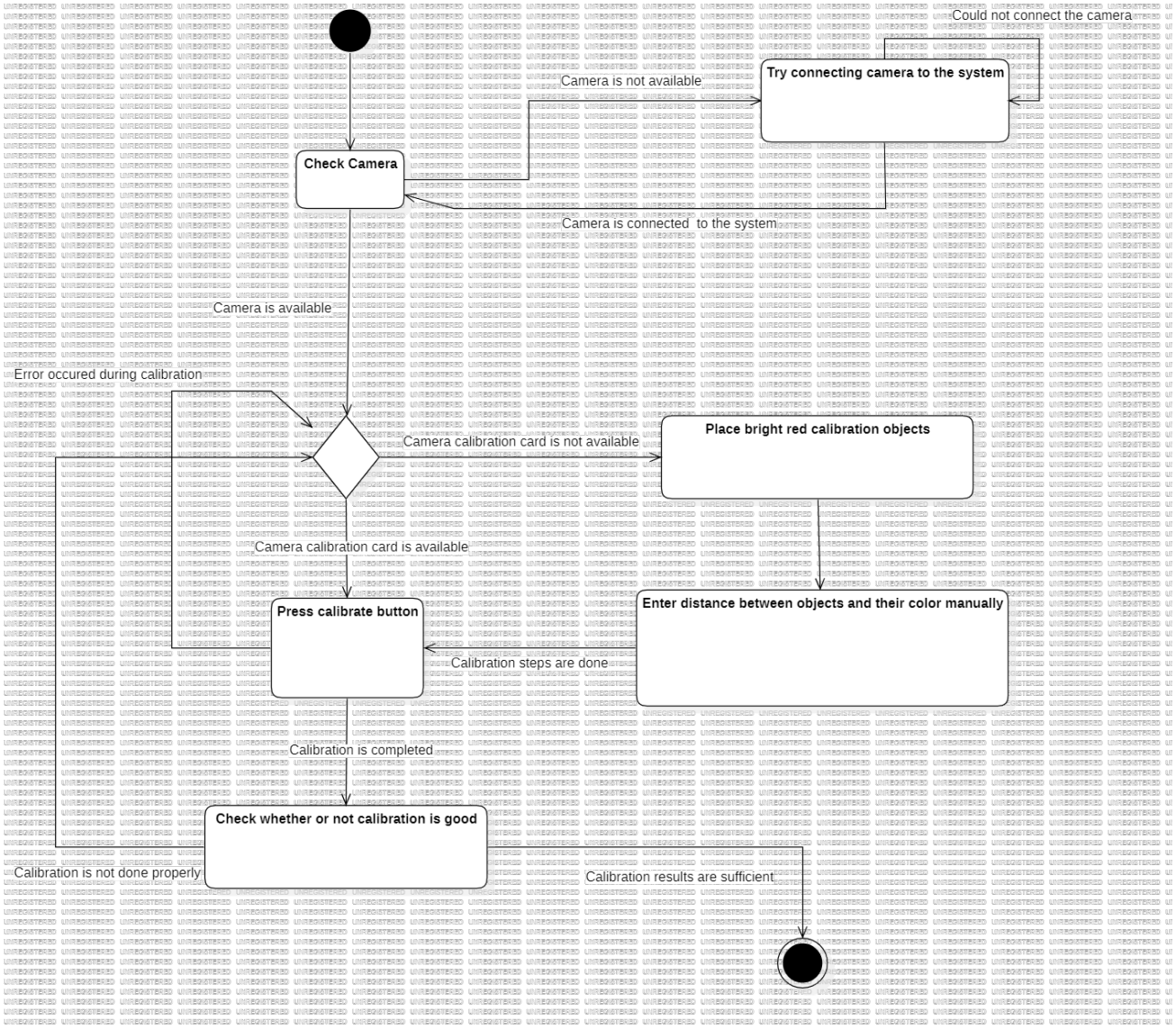

Figure 3.3: Activity Diagram for Calibrate Camera Use Case

Use-Case Name	View Photos
Actors	User
Description	The user views past photos
Data	Photos
Preconditions	1. User must be connected to the system. 2. Photos should be available

Stimulus	The user clicks on "Photos" section
Basic Flow	Step 1 – User goes to the "Photos" section in Web App Step 2 - Photos are shown to user and the process is done.
Alternative Flow#1	Step 1 – User goes to the "Photos" section in Web App Step 2 - Since there aren't any photos taken previously, It says "You haven't yet taken any photos with your FarmBot. Once you do, they will show up here".
Alternative Flow#2	-
Exception Flow	-
Post Conditions	Photos are viewed by the user.

Table 3.3: View Photos

Use-Case Name	Add Regimen
Actors	User
Description	The user adds new regimen
Data	Sequences
Preconditions	1. User must be connected to the system. 2. There must be some previously added sequences.
Stimulus	The user clicks on "Regimen" panel

Basic Flow	Step 1 - User goes to the "Regimes" section in Web App Step 2 - User decides on how many week that regime should continue Step 3 - User selects the duration that the regime should be applied Step 4 - User selects the sequences to use in that regime Step 5 - New regimen is added
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no previously added sequences, user must first create one.
Post Conditions	New regimen must be added.

Table 3.4: Add Regimen

Use-Case Name	Add Sequence
Actors	User
Description	The user adds new sequence
Data	Commands
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on "Sequences" section

Basic Flow	Step 1 – User goes to the "Sequences" section in Web App Step 2 – User clicks on "+" button to add new sequence Step 3 - User adds a description to the sequence. Step 4 – User adds as many variables as he/she want Step 5 - Using predefined variables, user adds sequence steps using commands like move, if...else etc. Step 6 – User presses save button and the new sequence is added.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	New sequence must be added.

Table 3.5: Add Sequence

Use-Case Name	Verify Email
Actors	User
Description	Email entered by user is verified by sending verification mail
Data	Email
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on "Create Account" button

Basic Flow	Step 1 - After user presses the "Create Account" button, email verification email is sent to the user's mail. Step 2 - User clicks on the link sent by the mail. Step 3 - Email verification is done.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	Email is verified.

Table 3.6: Verify Email

Use-Case Name	Add Curve
Actors	User
Description	The user adds new curve
Data	Plot points
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on add button in "Curves" section
Basic Flow	Step 1 – User goes to the "Curves" section in Web App Step 2 - Water/Spread/Height curve options are shown to the user. Step 3 - User chooses to add water curve. Step 4 - User can choose different templates to create the curve. In addition to the default templates, user can specify his/her own plot by specifying points. Step 5 - After the modifications are done, "curve adding" process is done, because system automatically saves the curves.

Alternative Flow#1	Step 1 – User goes to the "Curves" section in Web App Step 2 - Water/Spread/Height curve options are shown to the user. Step 3 - User chooses to add spread curve. Step 4 - User can choose different templates to create the curve. In addition to the default templates, user can specify his/her own plot by specifying points. Step 5 - After the modifications are done, "curve adding" process is done, because system automatically saves the curves.
Alternative Flow#2	Step 1 – User goes to the "Curves" section in Web App Step 2 - Water/Spread/Height curve options are shown to the user. Step 3 - User chooses to add height curve. Step 4 - User can choose different templates to create the curve. In addition to the default templates, user can specify his/her own plot by specifying points. Step 5 - After the modifications are done, "curve adding" process is done, because system automatically saves the curves.
Exception Flow	-
Post Conditions	New curve must be added to the system.

Table 3.7: Add Curve

Use-Case Name	Install Farmbot OS
Actors	User, Raspberry Pi Imager

Description	The user installs the Farmbot OS to the Raspberry Pi, using Raspberry Pi Imager software
Data	Farmbot OS image file
Preconditions	-
Stimulus	The user proceeds through Farmbot setup
Basic Flow	Step 1 - User downloads the Farmbot OS image file using Web App. Step 2 - User downloads and installs the official Raspberry Pi Imager program from the Raspberry Pi Foundation. Step 3 - Using Raspberry Pi Imager software, user writes the Farmbot OS image file to the microSD card. Step 4 - User inserts the microSD card into Raspberry Pi and the installation process is done.
Alternative Flow#1	-
Exception Flow	-
Post Conditions	Farmbot OS is downloaded to the Raspberry Pi.

Table 3.8: Install Farmbot Os

Use-Case Name	Setup Product
Actors	User
Description	The user setup the product
Data	FarmBot
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on setup button in main screen

Basic Flow	Step 1 - The user specifies Farmbot's "Order Number", preferred language and many other things that are needed. Step 3 - The user downloads the Farmbot OS using Raspberry Pi imager (Details are in another use case called "Install Farmbot Os"). Step 4 - User completes the configuration steps written in setup.farm.bot. Step 5 - User then manually configures map, motors, movements, camera, tools etc. Step 2 - After completing all configuration steps, Farmbot setup is completed.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If user presses "No" button in any step of setup, there appears three option: 1. I do not wish to continue yet: - If this is the case then App says to the user "Press YES when ready". 2. I pressed the wrong button: -If this is the case then App says to the user "Press YES". 3. Something else happened and I need additional help -If this is the case then App shows an feedback form to the user, where he/she can submit what's wrong with the user.
Post Conditions	-

Table 3.9: Setup Product

Use-Case Name	Add Plant
Actors	User, OpenFarm
Description	The user add new plant to the farm
Data	Plot points, Information about plant
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on add button in "Plants" section
Basic Flow	Step 1 – User goes to the "Plants" section in Web App Step 2 - User clicks to "add plant" button. Step 3 - User types the plant name and chooses from the alternatives. Step 4 - After the selection of plant, user can click on "+grid" button to add the plant to the farm. Step 5 - User chooses the locations where that specific plant will be planted. Step 6 - User can also specify the curves(water, spread, height) that will be applied to the plant or the start date that the plant will be planted. Step 7 - After specifying all of these labels, process is done.
Alternative Flow#1	-
Exception Flow	At Step 4, if the plant doesn't exist in OpenFarm database then, many information will be missed about the plant. But that doesn't mean it cannot be added.
Post Conditions	New plant must be added to the system.

Table 3.10: Add Plant

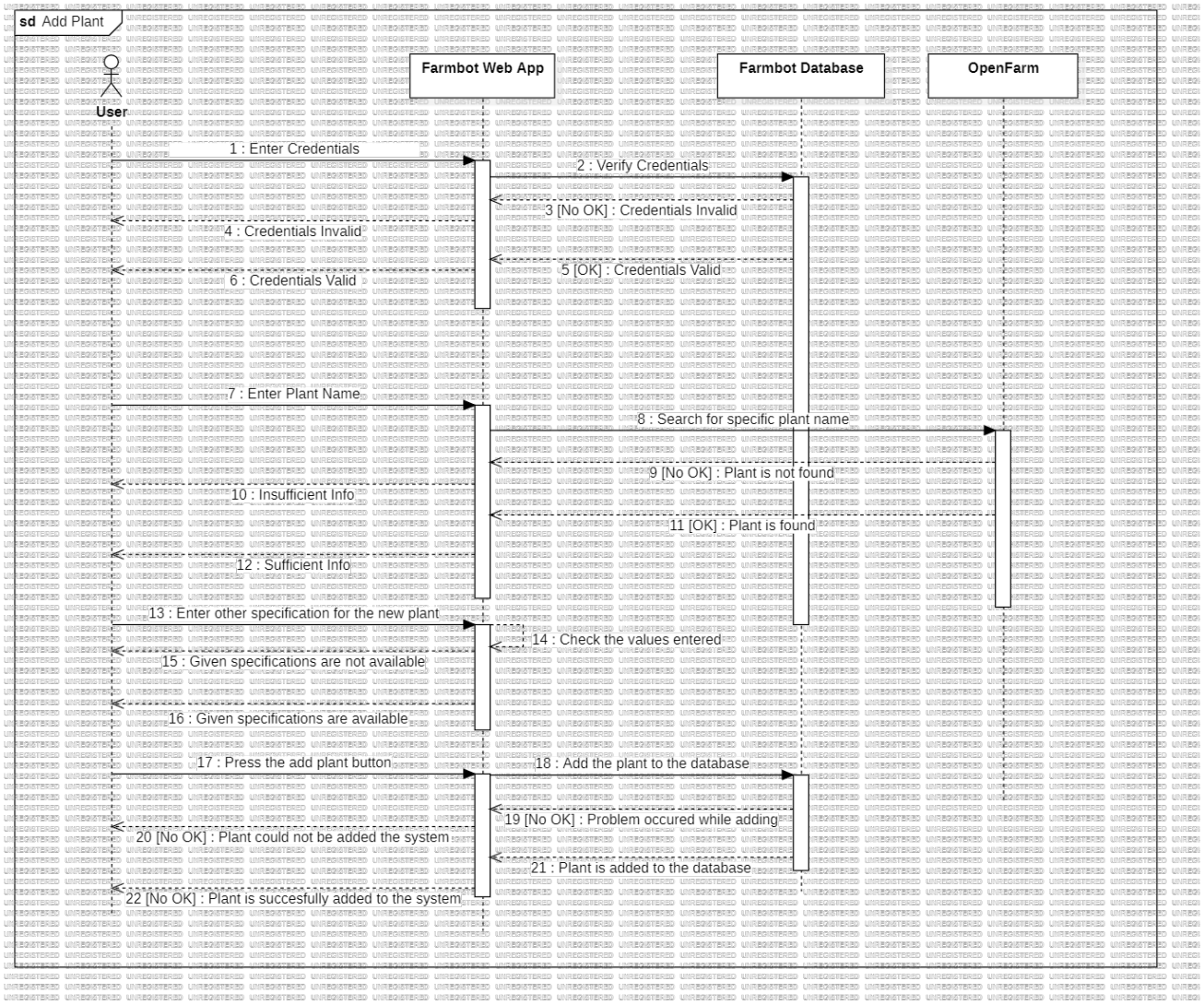

Figure 3.4: Sequence Diagram for Add Plant Use case

Use-Case Name	Login
Actors	User
Description	The user login to the FarmBot system.
Data	Credentials
Preconditions	-
Stimulus	User wants to login.

Basic Flow	Step 1 - User fills up the login credentials. Step 2 - User clicks to the "Login" button. Step 3 - System checks the credentials and logs the user in if they match with the ones with the database and the process is done after that.
Alternative Flow#1	Step 1 - User realizes that he/she forgot the login credentials. Step 2 - User clicks to the "Forgot password?" button. Step 3 - User enters his/her email to the "Password Reset" form. Step 4 - System sends an password reset link to user. Step 5 - User decides on a new password. Step 6 - User can login with the new credentials. The process is done.
Exception Flow	- If the credentials doesn't match with the ones in the database, then the user gets error.
Post Conditions	User logs in to the Web App.

Table 3.11: Login

Use-Case Name	Create Account
Actors	User
Description	The user sign up to the FarmBot system.
Data	Plot points
Preconditions	1. User must be connected to the website.
Stimulus	User wants to register

Basic Flow	Step 1 - User fills up the registration form with their details. Step 2 - User checks the "I agree to the Privacy Policy and Terms of Use" box Step 3 - User clicks to "Create Account" button. Step 4 - System registers the user and the process is done after that.
Alternative Flow#1	-
Exception Flow	- If the verification password doesn't match with the first written password, then the user gets error. - If the written email is not a valid email then user gets error.
Post Conditions	User is registered to the system

Table 3.12: Create Account

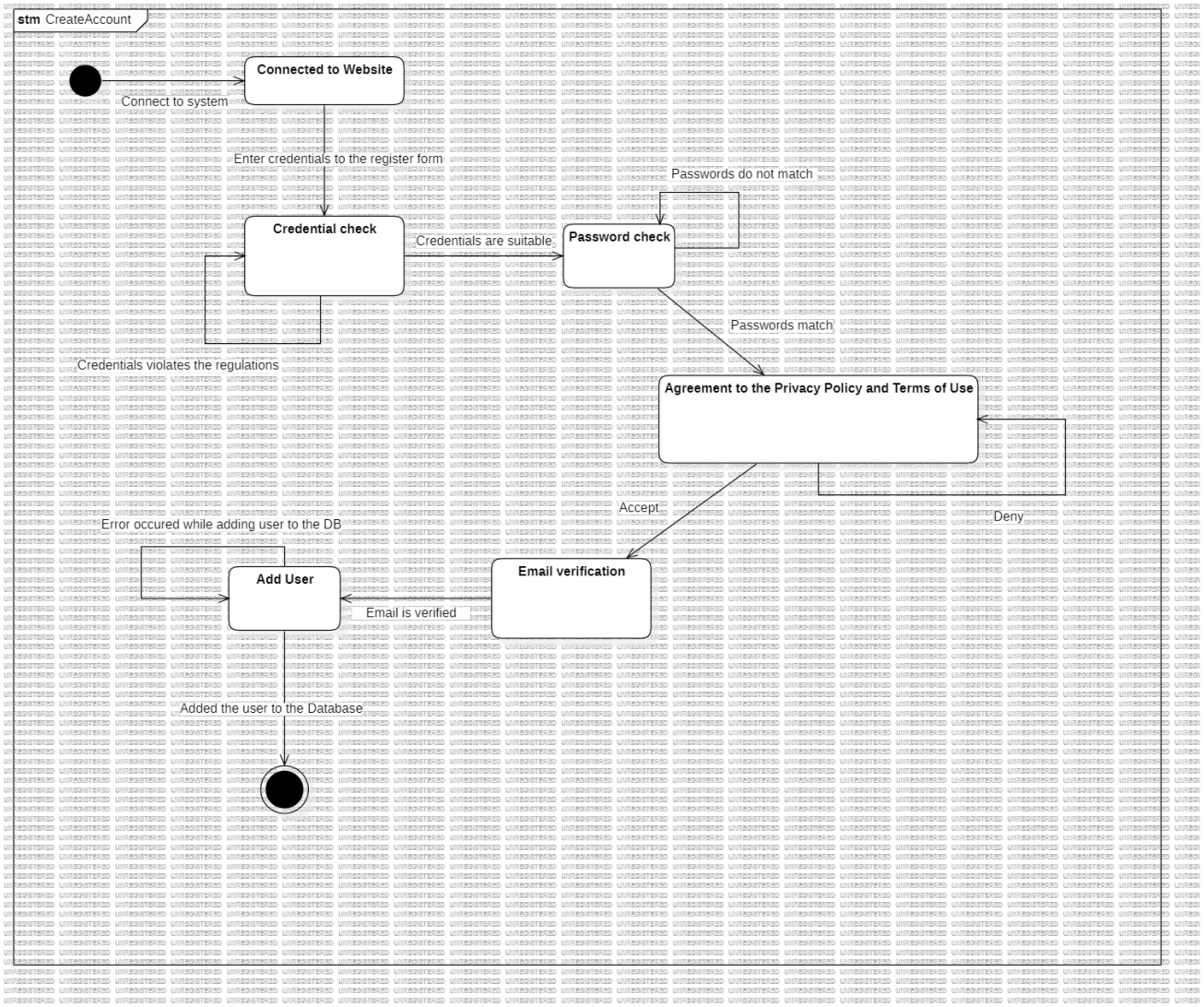

Figure 3.5: State Diagram for Create Account Use case

Use-Case Name	Read Sensor
Actors	User
Description	The user chooses a sensor to read
Data	Sensor's Output
Preconditions	1. User must be connected to the system. 2. Sensor must be available
Stimulus	The user clicks on "Read Sensor" button

Basic Flow	Step 1 – The user clicks on "Read Sensor" button under "Sensors" section. Step 2 – The user selects the sensor he/she want to read. section. Step 3 - The user send request to read a sensor. Step 3 - The system gets the output of sensor when the user pressed the button. Step 4 - The system shows the reading to the user.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the sensor isn't available, user gets error.
Post Conditions	Sensor's output is read by the user.

Table 3.13: Read Sensor

Use-Case Name	Add Sensor
Actors	User
Description	The user adds a sensor to the Farmbot
Data	Sensor
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on "Add Sensor" button
Basic Flow	Step 1 – The user clicks on "Edit" button under "Sensors" section. Step 2 – The user then presses "+" button to add a new sensor. Step 3 - The user will specify the pin where the sensor is connected and whether or not the sensor is digital/analog. Step 4 - After the specifications, user presses save button and the process is done..

Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	New sensor is added to the Farmbot.

Table 3.14: Add Sensor

Use-Case Name	Add weed
Actors	User
Description	The user adds a weed to the Farmbot
Data	Sensor
Preconditions	1. User must be connected to the system.
Stimulus	The user clicks on "+" button to add weed
Basic Flow	Step 1 – The user clicks on "+" button to add weed, which is under "Weeds" section. Step 2 – The user then presses "+" button to add a new sensor. Step 3 - The user will specify (x,y,z) location of the weed and the radius of the weed. Step 4 - After the specifications, user presses save button and the process is done.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	New weed is added to the system.

Table 3.15: Add weed

3.3 Logical Database Requirements

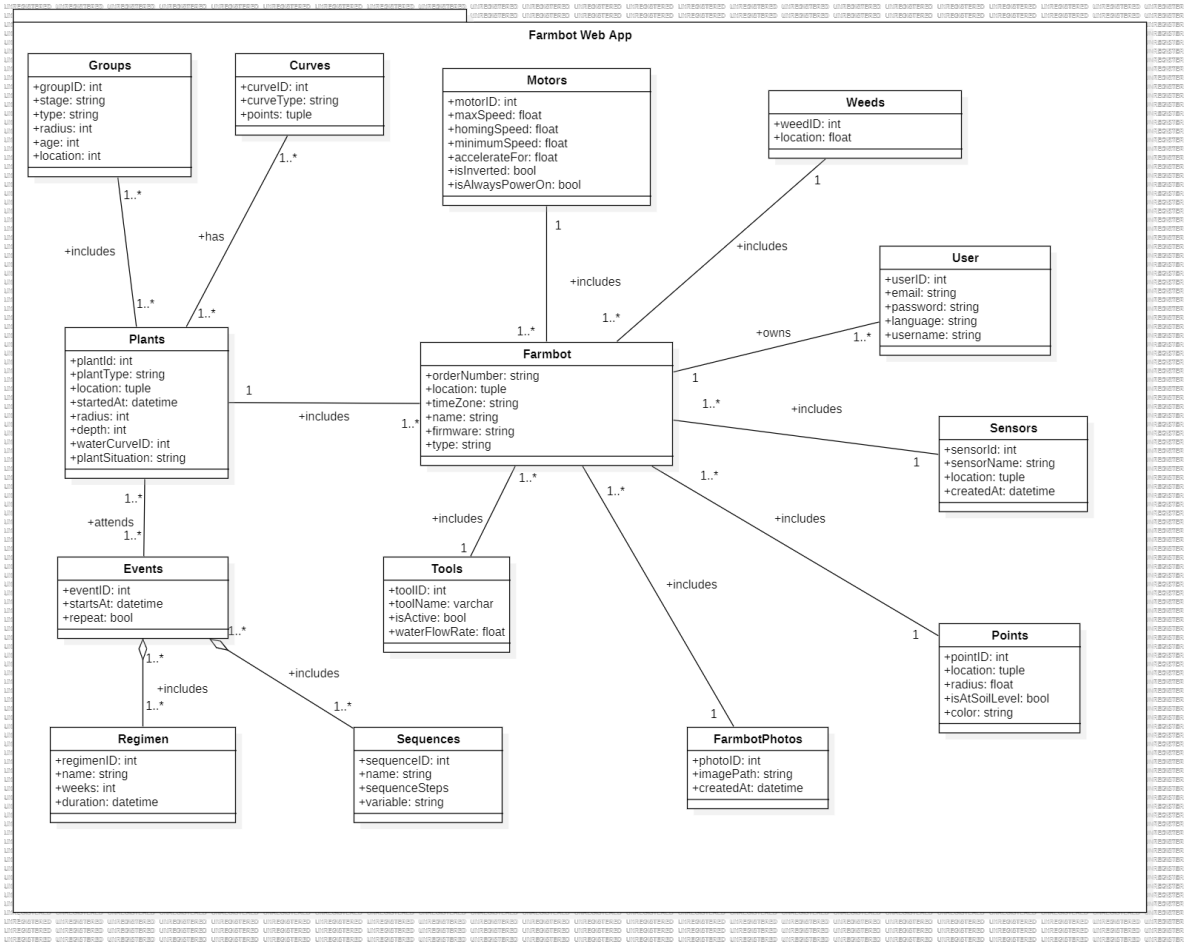

Figure 3.6: Class Diagram For Logical Database Representation

3.4 Design Constraints

- All saved data must adhere to the rules and regulations. For instance, privacy law must be followed in the system's design, while dealing with the users data.
- The system's storage, which stores the any piece of data, is confidential and needs to be maintained with utmost care. Disk level encryption is required to guarantee the security of data.

- Backups of the database and any storage device containing data, such as images or historical sensor readings, should be performed on a weekly basis. Using cloud backup is one possibility.

3.5 System Quality Attributes

3.5.1 Security

There are many safety features have been engineered into the FarmBot to provide users a safer experience and help mitigate various hazards.

- A physical E-stop button is located on top of the electronics box and can be pressed to immediately stop and unpower all motors and peripherals. This can be used in any emergency situations.
- FarmBot can detect motor stalls and will by default E-stop if a movement error occurs too many times in a row. This can prevent injury and potential damage to the machine.
- A fuse is located on the Farmduino to protect the electronics from over-current conditions that may be caused by a short or other malfunction.
- FarmBot can be configured to set a peripheral to a certain state after a timeout period. This can act as a secondary precaution in case a poorly designed sequence or custom code instructs FarmBot to leave a peripheral in an unsafe state for an extended period of time, such as leaving the solenoid valve open for too long.

3.5.2 Usability

- All functionalities of the system shall be ready to use for the user when they they are connected to the Web App.
- System should not be too complex. Even people who are not very proficient in computer use should be able to use the application easily.

- The system functionalities itself shall be self-explanatory, there shall not be any question marks in the head of the users. If there are some complex functionalities, side notes should be written for that.
- All the interfaces in the system shall have user-friendly views, they shall include lively colors and large enough fonts for everyone to see.

3.5.3 Performance

- Number of simultaneous users to be supported should be at least 1000 users.
- In less than a second, 95% of the tasks shall be completed.
- The software should be able to process and store up to 20 gigabyte of data per user, including text, images, and sensor data.

3.5.4 Reliability

- The electronic parts of the system should be checked regularly in case of an electronic shortcut or a failure. Regular tests for the Farmduino Express microcontroller, and for the Raspberry Pi Zero 2 W computer should have be applied.
- The system shall be tested if any component or module gets changed, updated, upgraded or if a new component gets installed.
- The health of the SD Card must be checked regularly to not to have a data loss failure.
- There should not be any data loss in the database.

3.5.5 Portability

- Users should be able to use FarmBot Web Application from a web browser using laptop, tablet, smartphone or any other device that is connected to Internet.
- The software application should be compatible with all Operating Systems. It shall run on MacOS, Windows, Linux, Android, iOS, and so on.

- The choice of the programming language shall not be platform-dependent.
- Any external libraries used in the system shall have up-to-date support and they shall work on all environments.

3.5.6 Availability

- The downtime of the application shall be less than 0.001% in a year.
- The system backup process shall be done at a time when the application usage statistics are at their lowest.
- If the system needs to be restarted or updated, then the system shall be available again in at most 10 minutes.

3.5.7 Maintainability

- Every three months, users shall inspect their hardware for loose screws or visible damage such as corrosion, bending, and cracking of components.
- Integration of new sensors or tools shall not cause to an error.
- Documentation of system shall be updated regularly and shall help to use of system.
- Integration of the new system shall not lead to the crash of the application.

3.6 Supporting Information

- FarmBot is an open source precision agriculture CNC farming project consisting of a Cartesian coordinate robot farming machine, software and documentation including a farming data repository. Since it is an fully Open-Source project, It is developing day by day with the contribution of people.

- Traditional farming methods often involve labor-intensive tasks such as planting, watering, and weeding, which require significant time and effort. FarmBot automates these tasks, reducing the need for manual labor and enabling farmers to focus on strategic decision-making and farm management.

- Efficient resource utilization is crucial for sustainable agriculture. And FarmBot optimizes the use of resources such as water, fertilizer, and energy by delivering them precisely where and when needed based on plant requirements. This optimization reduces waste and environmental impact.

4. Suggestions to Improve The Existing System

4.1 System Perspective

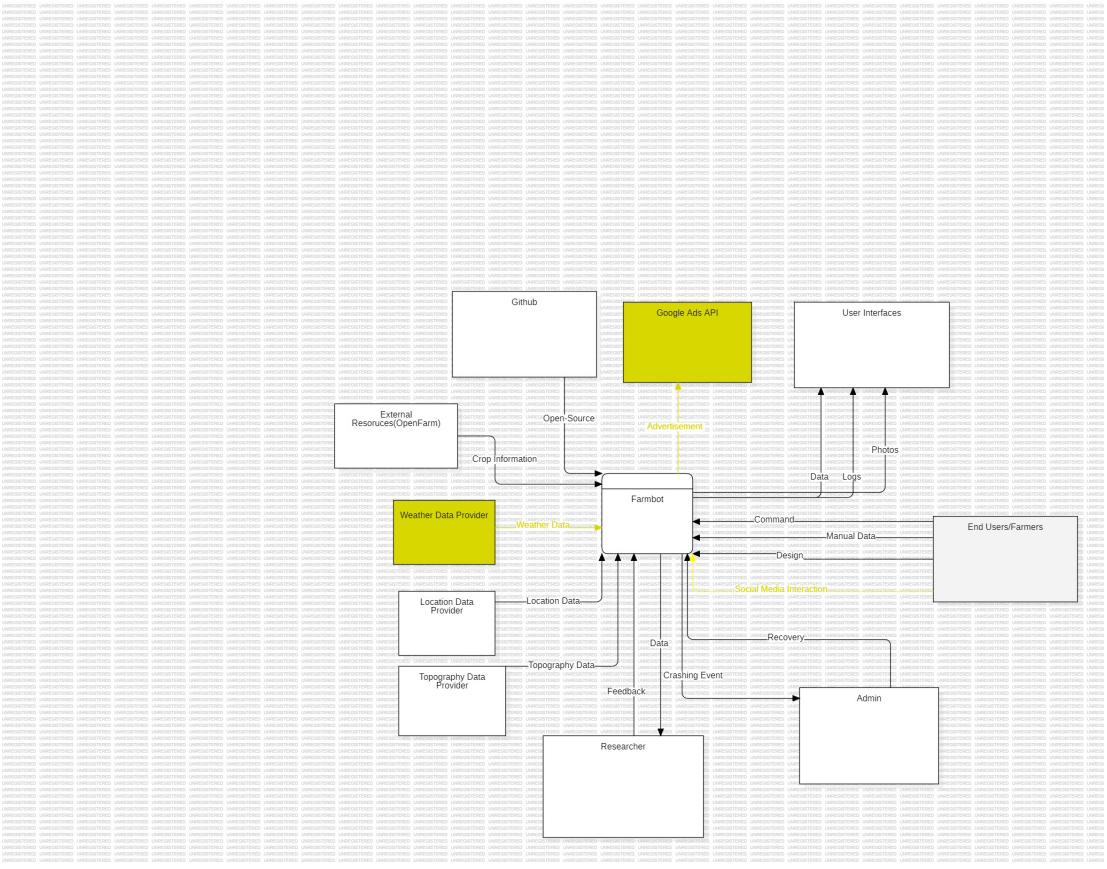

Figure 4.1: Context Diagram

Newly introduced/suggested external entities and data flows:

- The system will use weather data in order to predict much better with Decision Support System. This data will be integrated with the location data for correct weather data retrieval. The data source can vary location to location. For example, we thought that using OpenWeather API as an weather data source, would beneficial. Because it provides data globally. But in case of OpenWeather API being insufficient, We thought that it would be beneficial to use official data providers for each country. For instance, Turkey's most popular data provider is "Meteoroloji Genel Mudurlugu", or official weather data provider for USA is National Weather Service etc.

ANLIK DURUM (05 Nisan - 21.01)

13,7°C Parçalı Bulutlu | Yağış 0mm | Nem %31 | Rüzgar 14km/sa | A. Basınç 915,8hPa | D.I.Basınç 1015,5hPa

SAATLİK TAHMİN

Saat	Beklenen Hadise	Sıcaklık (°C)	Hissedilen Sıcaklık (°C)	Nem (%)	Rüzgar Yönü	Ort. Rüzgar Hızı (km/sa)	Maks. Rüzgar Hızı (km/sa)
CUMA 21.00 - 24.00		9	9	67		7	18
CUMARTESİ 24.00 - 03.00		6	6	77		6	13
CUMARTESİ 03.00 - 06.00		4	4	84		6	12
CUMARTESİ 06.00 - 09.00		10	10	53		6	15
CUMARTESİ 09.00 - 12.00		15	15	34		11	27
CUMARTESİ 12.00 - 15.00		18	18	26		13	29
CUMARTESİ 15.00 - 18.00		17	17	27		7	29
CUMARTESİ 18.00 - 21.00		11	11	50		6	15
CUMARTESİ 21.00 - 24.00		9	9	67		6	13

Figure 4.2: Example Weather Data

- The company will use Google Ads API in order to advertise their product and web app. Since the system is not popular world-wide, the company need to introduce their product in order to make more profit.

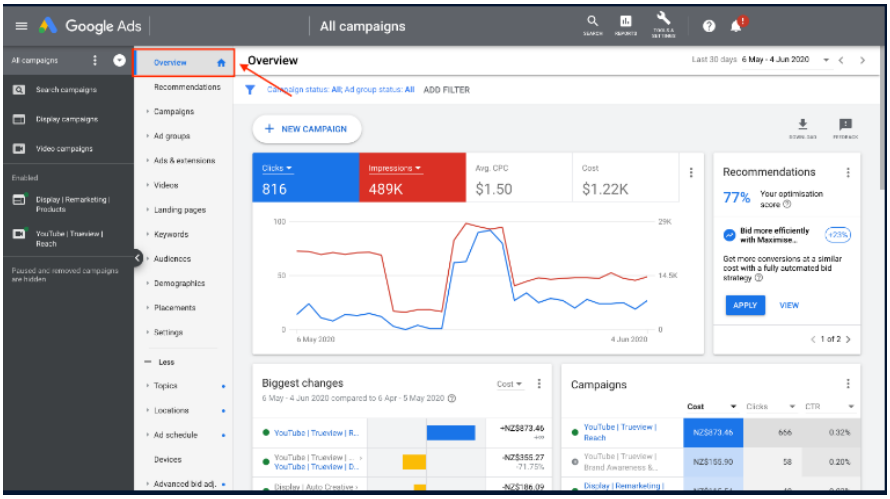

Figure 4.3: Google Ads Interface

- A new social media platform integrated in the Farmbot Web App. New data flow added as Social Media Interaction, since only the Farmbot users can use this platform.

4.2 External Interfaces

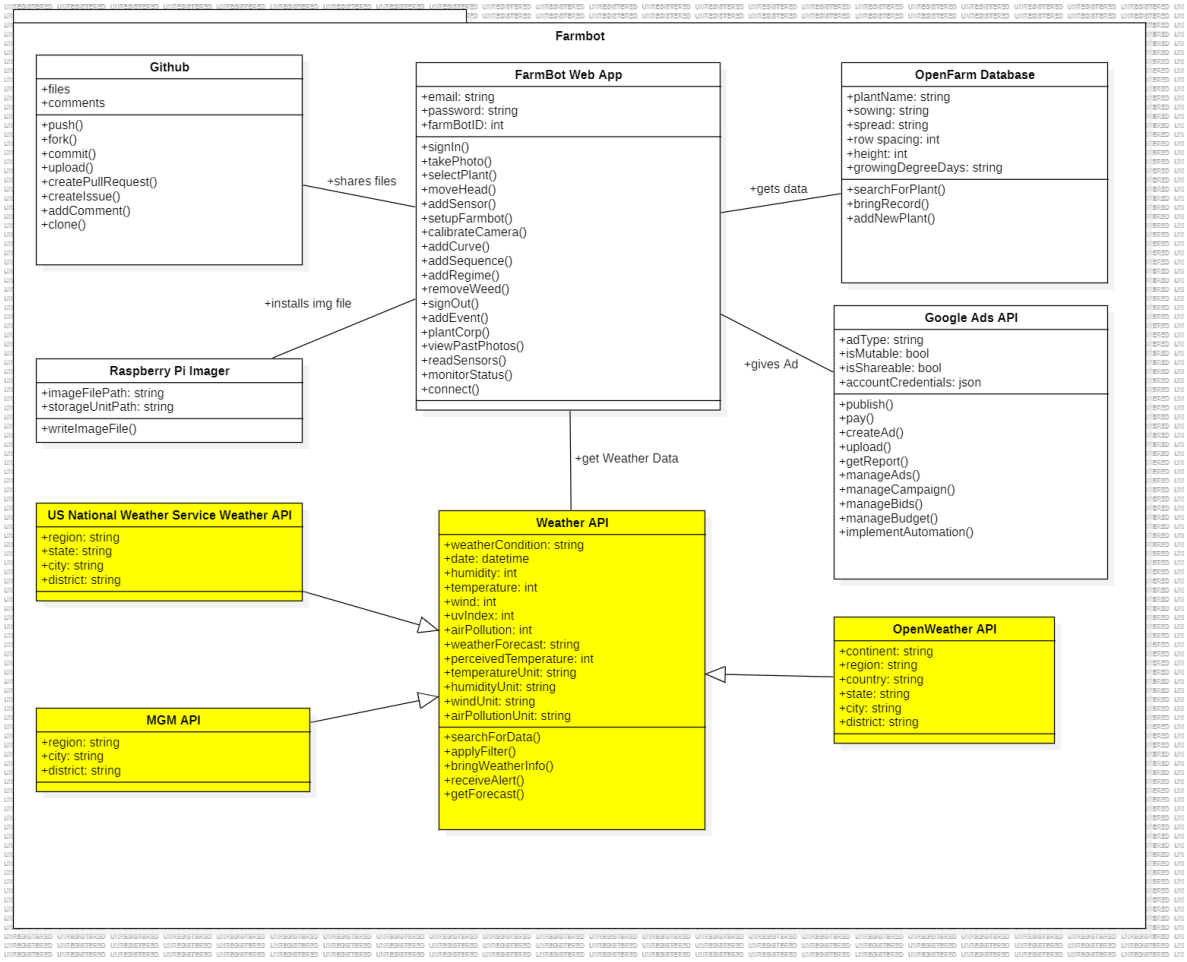

Figure 4.4: Recommended External Interfaces

- **Weather API:** The weather data provider can vary location to location. So there is not any certain interface for any region. Each source will be chosen according to past success. If there is not any data provider for a region, OpenWeather will be chosen by default. This suggestion aims to improve Decision System's accuracy and maximize productivity.

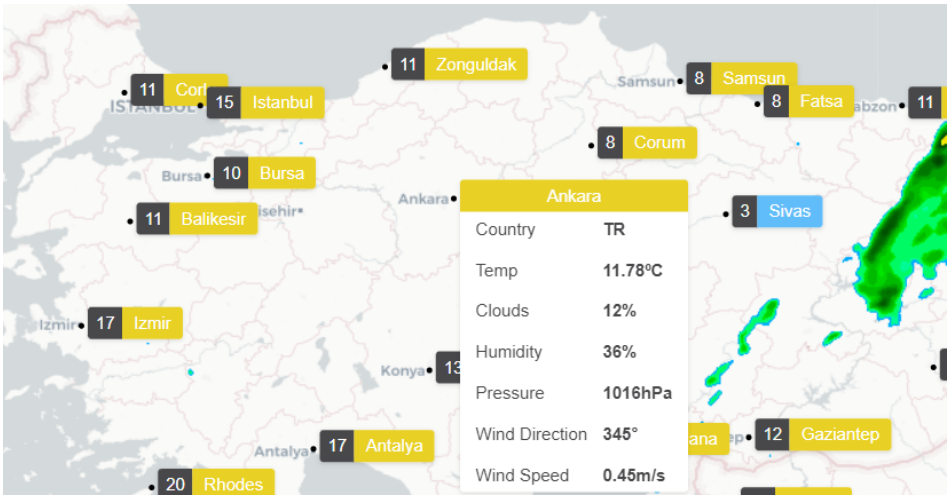

Figure 4.5: OpenWeather Interface

- **Google Ads API:** The Farmbot is not very popular world-wide. It needs to be recognized and advertised. More customers mean more financial resources, and these resources mean that more improvement can be made into the system. This advertisement process can also attract researchers and developers around the world, and can increase quality of the team. More people need to know about Farmbot!

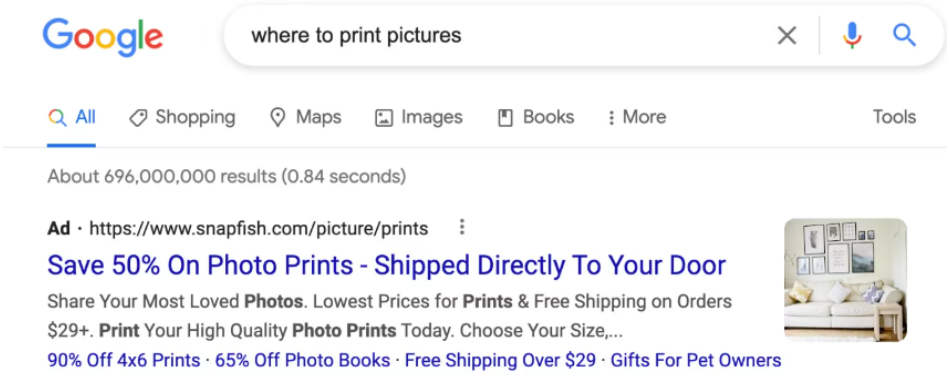

Figure 4.6: Sample Google Ad

4.3 Functions

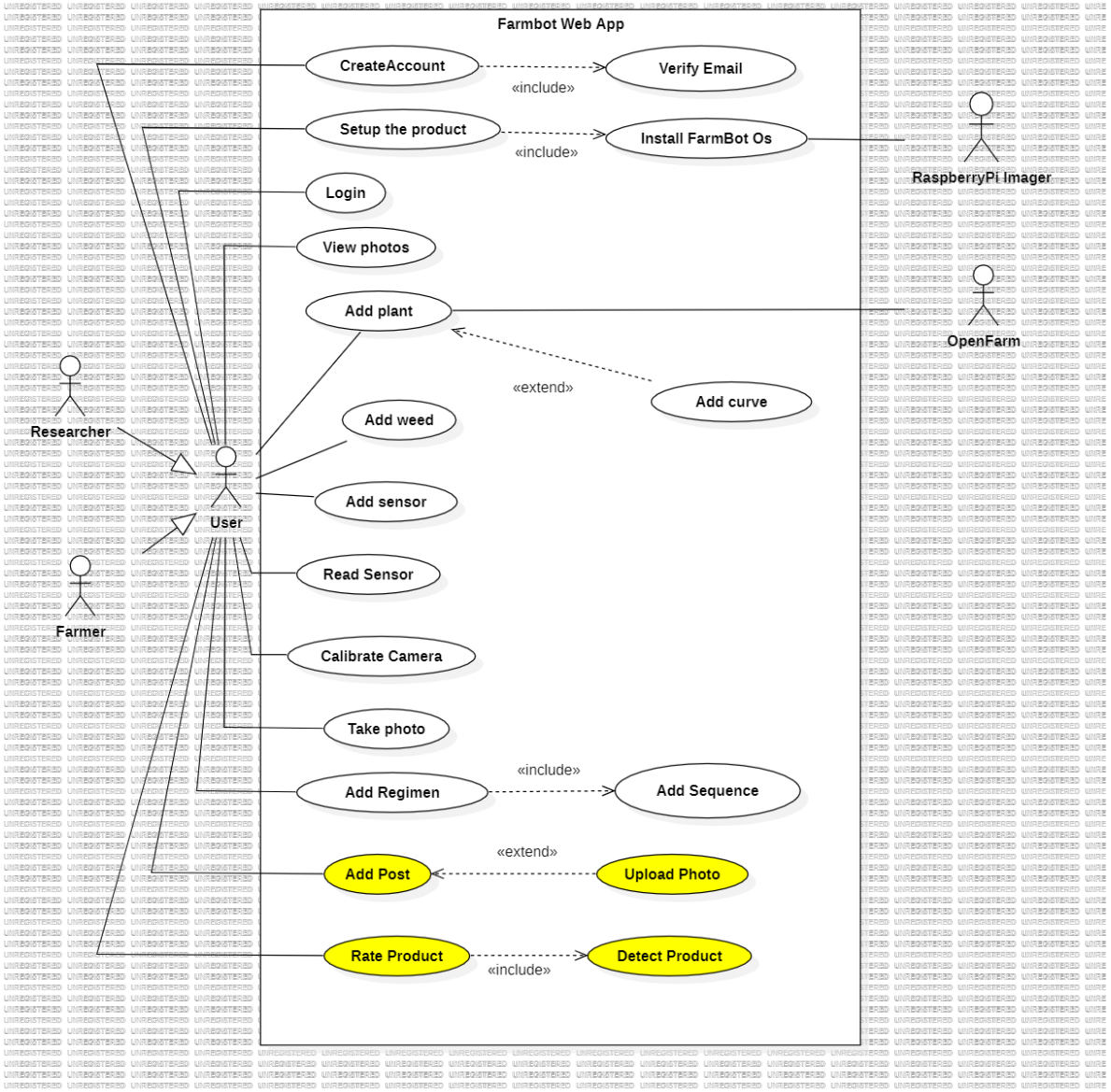

Figure 4.7: Use case diagram for recommendations

Use-Case Name	Rate Product
Actors	User

Description	The user rates the products using the installed hardware
Data	Product
Preconditions	1. User must be connected to the system.
Stimulus	The user presses button called "Detect".
Basic Flow	Step 1 - The user presses button called "Detect" to detect the products using computer vision and the camera. Step 2 - Then user selects one of the alternative products that are detected by camera. Step 3 - User presses the button called "Rate". Step 4 - Using the hardwares that will be installed to the system, the report is shown to the user and its ranking among the other products that are previously grown by the users.
Alternative Flow#1	Step 1 - The user presses button called "Detect" to detect the products using computer vision and the camera. Step 2 - If camera couldn't detect the products, user can manuelly move the camera and label the product. Step 3 - User then presses the button called "Rate". Step 4 - Using the hardwares that will be installed to the system, the report is shown to the user and its ranking among the other products that are previously grown by the users.
Exception Flow	-
Post Conditions	New product and its rate is added to the system.

Table 4.1: Rate Product

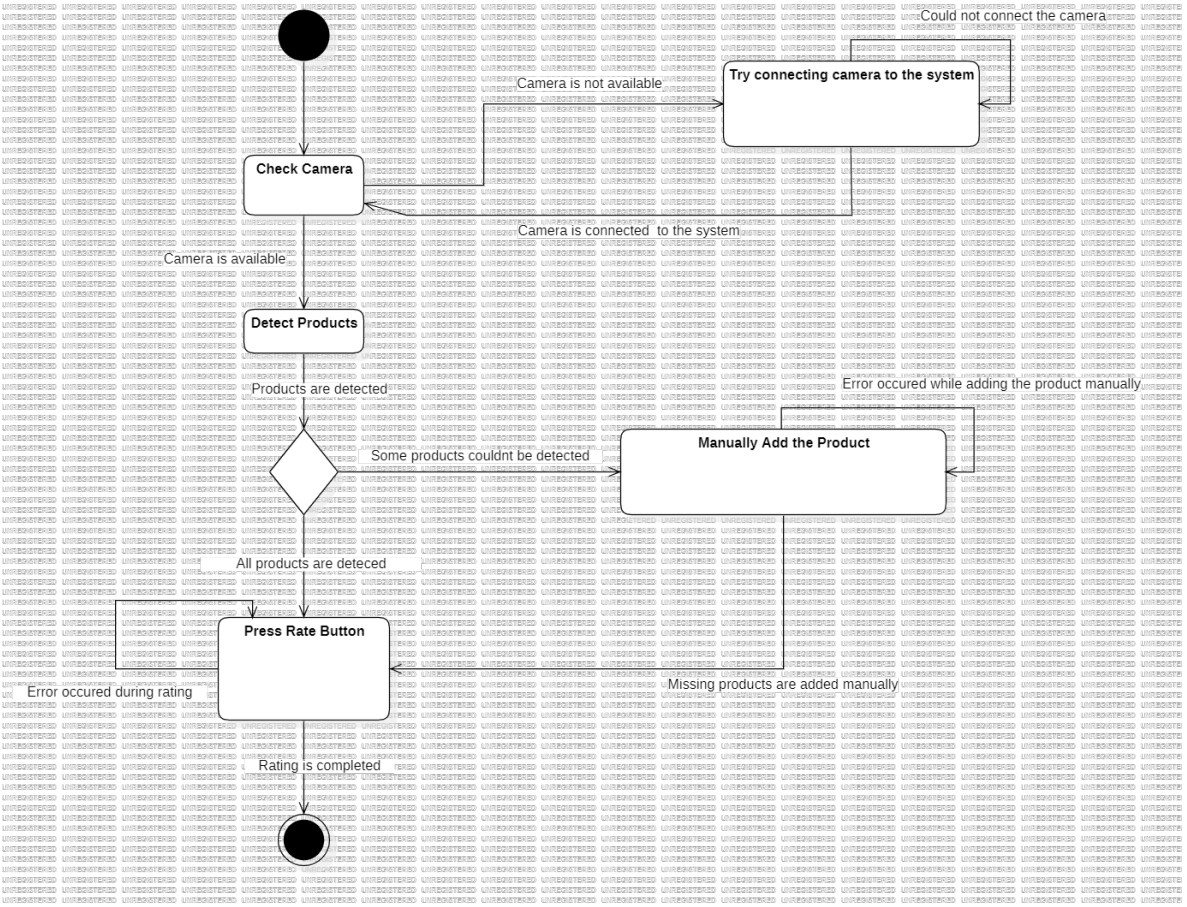

Figure 4.8: Activity Diagram for Rate Product Use Case

Use-Case Name	Detect Product
Actors	User
Description	The user clicks the ”Detect” button
Data	Photo
Preconditions	1. User must be connected to the system.
Stimulus	The user presses button called ”Detect”.
Basic Flow	Step 1 - The user presses button called ”Detect”. Step 2 - Using pre-trained computer vision model, products are detected. Step 3 - Found products are shown to the user.

Alternative Flow#1	Step 1 - The user presses button called "Detect". Step 2 - Using pre-trained computer vision model, products are detected. Step 3 - If there exists a product that couldn't be detected by the algorithm, user can label the product by moving the camera to that location. Step 4 - Found/labeled products are shown to the user.
Exception Flow	-
Post Conditions	Products that are grown in FarmBot are shown to the user.

Table 4.2: Detect Product

Use-Case Name	Upload Photo
Actors	User
Description	The user attach a photo to the newly added post
Data	Photo, Post
Preconditions	1. User must be connected to the system. 2. User should have added a post
Stimulus	The user presses button called "Upload".
Basic Flow	Step 1 - The user presses button called "Upload" to open the file browser window to attach the photo into post. Step 2 - Then user selects the photos from local or cloud storage units. Step 3 - User presses the button called "Finish" to end the process. Step 4 - If the photos uploaded don't exceed the size limit then they are attached.

Alternative Flow#1	-
Exception Flow	-
Post Conditions	Photo is attached to the post.

Table 4.3: Upload Photo

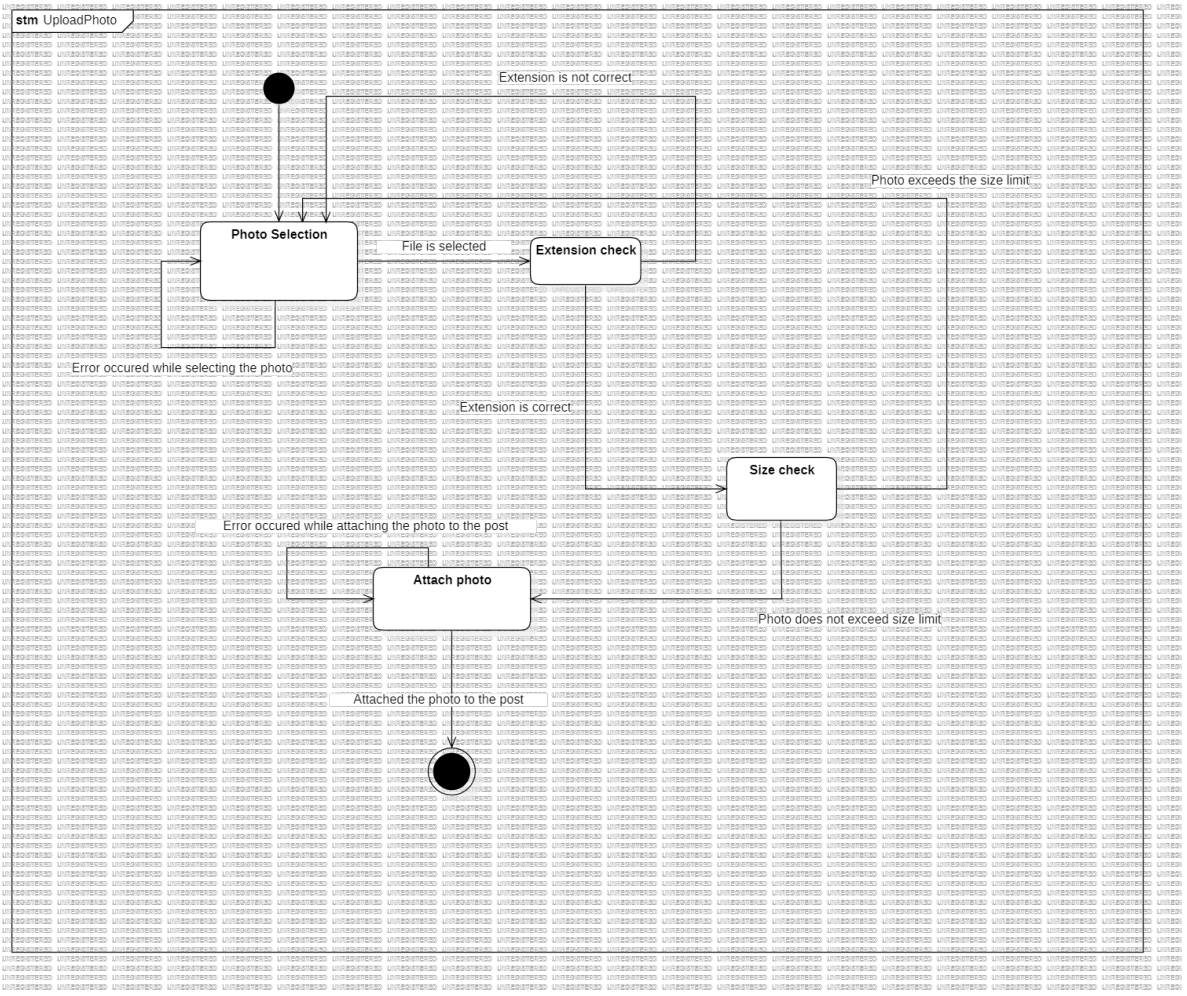

Figure 4.9: State Diagram for Upload Photo Use Case

Use-Case Name	Add Post
Actors	User
Description	The user adds a new post to the social forum
Data	Photo
Preconditions	1. User must be connected to the system.
Stimulus	The user presses button called "Add".
Basic Flow	Step 1 - The user presses button called "Add" to add new post to the forum. Step 2 - The user enters the post's message, and its title. Step 3 - Then user presses button called "Upload" to attach photos to the post, if he/she wants. Step 4 - The user then presses button called "Send", to add the post.
Alternative Flow#1	-
Exception Flow	-
Post Conditions	New post is added to the social forum.

Table 4.4: Add Post

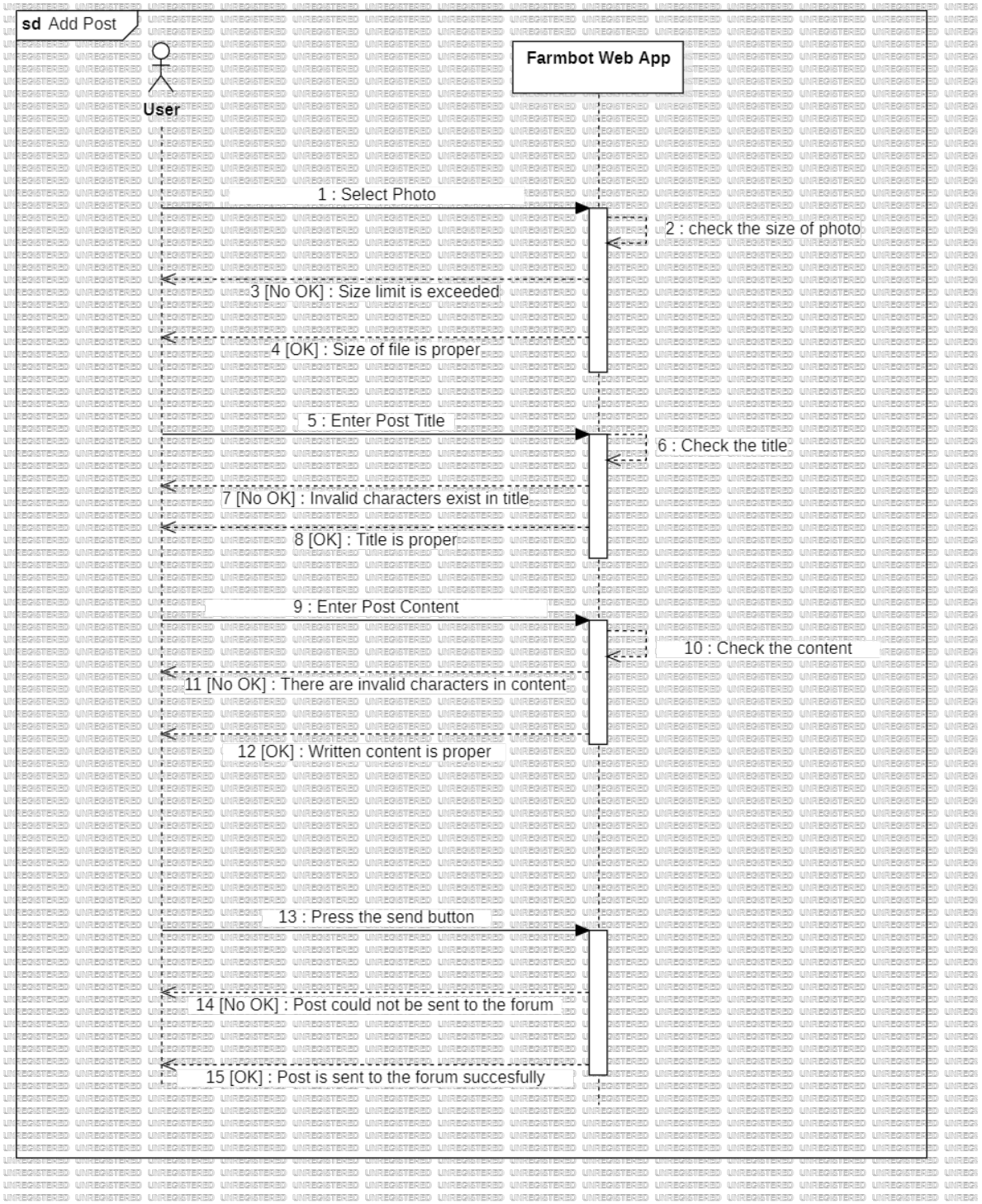

Figure 4.10: Sequence Diagram for Add Post Use Case

4.4 Logical Database Requirements

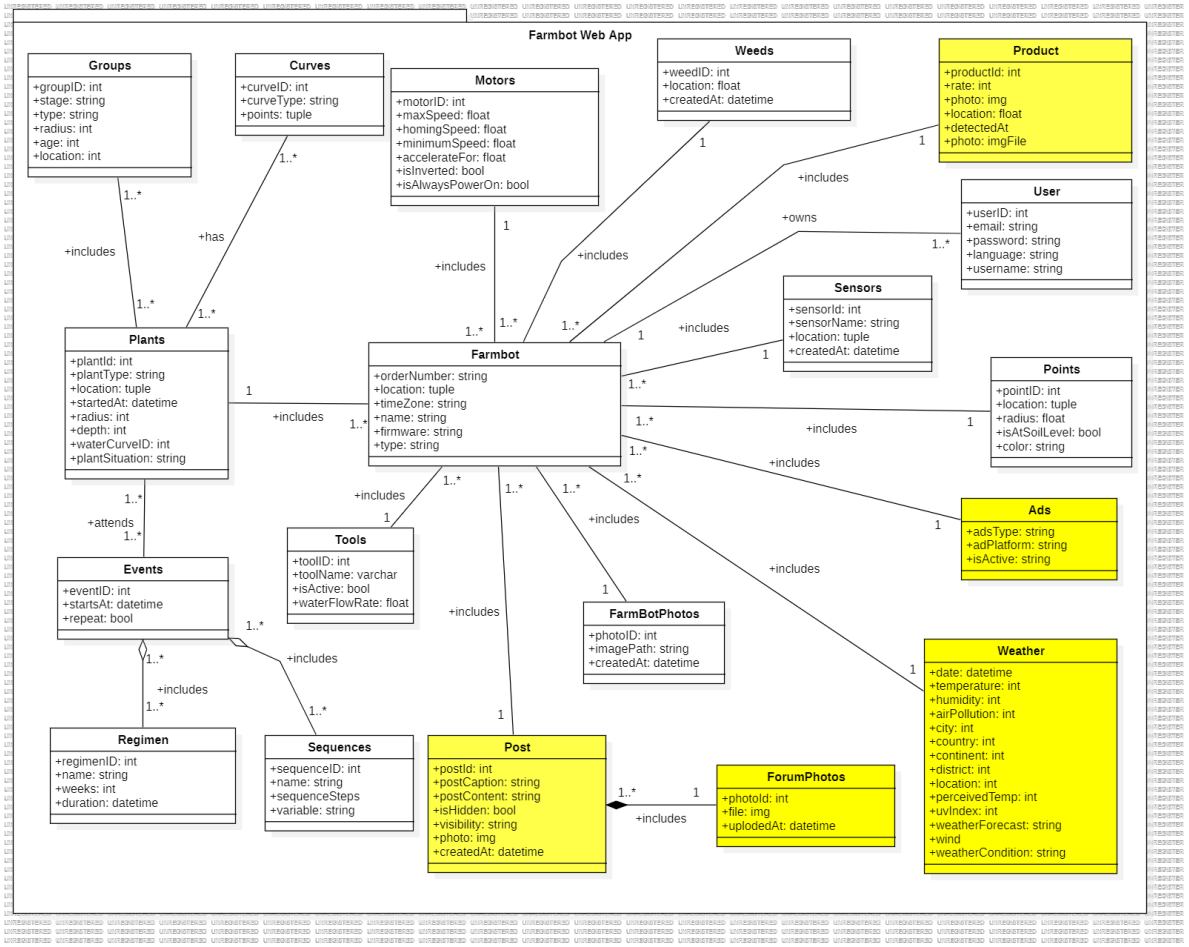

Figure 4.11: Class Diagram for Logical Database Requirements

4.5 Design Constraints

- Even though the hardware part of the system(instruction set, operating system etc.) seems like bug-free, always there can be problems and errors. In hardware point of view, any bug causes unintentional movement can damage other fields around user's field. These kind of errors can put the user in legally problematic situation.

- Any security issue can cause serious problems in terms of personal data security.

For example the system is using location data of the user for Decision Support System. Any leaking location data is the problem for the company. The company would possibly face some legal problems.

- Cost of the Farmbot product is also a major problem for company. Any fluctuation in raw materials' cost, transportation cost or a global economical crisis can put the company in trouble economically. Therefore materials used in Farmbot must be chosen cost-friendly as much as possible.

- Since Farmbot's hardware is ofted used outside, it should be resistant to weather conditions. It can be damaged or even damage the user(electric shock etc.). Thus the hardware system must be durable as much as possible.

- The company must investigate all the laws in any case of legal trouble for usage of Farmbot. If necessary, permission must be obtained from local and national governments.

- The Farmbot should be compatible with different types of field types. For example, the dynamics of a rice field and a corn field is different.

- The Farmbot must be agile and meet the requirements of the different type of users, and it can be very challenging for harsh field conditions. Designers may want to take such issues into consideration and design different versions of Farmbot for different conditions.

4.6 System Quality Attributes

4.6.1 Security

- Since new system has social media-like platform, security is more important. Every user will be identified by their profile, and in the case of any hacking or account stealing, the attacker may behave in unwanted way to damage user. Therefore security level must be increased.

- Since our new hardware will recognize different type of products, it must be in touch with the products, and this interaction may affect hardware. Therefore materials used should be safer and resistant to liquid.

4.6.2 Usability

- Every user should understand the social media platform easily, and it should attract user's attention. Current social media approaches should be used. - Hardware will perform operations automatically, but it must be cleaned up because of recognizing step. So all parts must be detachable and easy to clean.

4.6.3 Reliability

- New hardware component may break down easily because of interactions. It should be checked regularly for maximum functionality.

4.6.4 Portability

- New social media platform will be in Farmbot Web App, so if user can access Web App, he/she can also use social media platform. In other words it will have same portability attributes as Web app.

- Newly introduced hardware component will use same OS as original hardware, since it is integrated with that. Different parts will communicate with signalling.

4.6.5 Availability

- As stated, the social media platform will be in Web App, as long as Web App is reachable, social media platform will be reachable as well. In case of any crash specific to social media platform, the user will not reach it until recovery.

4.6.6 Maintainability

- New hardware is prone to break down, since it has lots of interaction with products such as tomato, watermelon etc. So it should be checked after every operation

from the system status and physically.

4.6.7 Performance

- The social media platform will not have more than web app user at a time. So it should be designed with the worst case in mind, which is all users are using social media at the same time, and it must has the capability of 1000 users hosting.
- Newly introduced hardware component must finish operations such as recognizing product, rating product etc. under 30 seconds.

4.7 Supporting Information

- Various types of data exchanged between the Farmbot system and its users, such as commands for planting, watering, and harvesting crops, as well as feedback on crop growth and health. The user should be aware of these kind of data and be able to interpret what these mean in order to maximize the throughput from the system.
- Cost analysis studies may include comparisons of manual farming methods versus automated farming using Farmbot, considering factors such as labor costs, water usage, and crop yields.
- Background information on agricultural practices, technological advancements in farming, and the rationale behind developing Farmbot can provide context for understanding its purpose and functionality. Since the most of the users are farmers, this background can be expected from them. Maybe a document or a tutorial can be supplied by the Farmbot company to users in order to minimize user level errors.
- The primary problems to be addressed by Farmbot include labor-intensive farming practices, inefficient resource utilization, and the need for sustainable agriculture solutions. By automating tasks such as planting, watering, and weeding, Farmbot

aims to reduce labor costs, optimize resource usage, and promote environmentally friendly farming practices.

- Farmbot is an open-soruce project! It means that all the code is available on its Github page. Even external developers may contribute to system and improve it. If the user can understand the rationale, he/she can use it more efficient. Check out the Github page!