TIANYI CHEN - 4095944

KEVIN VISSENBERG - 4059506

BRAM WIJSHAKE - 3467023

MARCO CARDILLO - 4076923

KRISTIYAN MANOLOV - 3605310

TOMA KRASTANOV - 3774678

Design Document Gardening System

Contents

[Design 3](#_Toc75105185)

[Problem 3](#_Toc75105186)

[Our solution 3](#_Toc75105187)

[Context Diagram 4](#_Toc75105188)

[Class Diagram 5](#_Toc75105189)

[Growbox: 5](#_Toc75105190)

[5](#_Toc75105191)

[Hardware 8](#_Toc75105192)

[Soil moisture sensor 8](#_Toc75105193)

[PH sensor 8](#_Toc75105194)

[Water level detector 8](#_Toc75105195)

[LDR sensor 8](#_Toc75105196)

[Distance sensor 9](#_Toc75105197)

[Heating 9](#_Toc75105198)

[Water pump 9](#_Toc75105199)

[Water valve 9](#_Toc75105200)

[Fan 9](#_Toc75105201)

[Grow light 10](#_Toc75105202)

[ESP32 10](#_Toc75105203)

[Water tubes 10](#_Toc75105204)

[Water tank 10](#_Toc75105205)

[Performance consideration 11](#_Toc75105206)

[Communication protocol 12](#_Toc75105207)

[Start Char = ‘$ ‘ 12](#_Toc75105208)

[Seperator Char = ‘: ’ 12](#_Toc75105209)

[End Char = ‘# ‘ 12](#_Toc75105210)

[Communication protocol description 17](#_Toc75105211)

[**Rich Shield Sensors** 17](#_Toc75105212)

[State Diagrams 19](#_Toc75105213)

[Watering system (water tank) 19](#_Toc75105214)

[Watering system (Grow box) 20](#_Toc75105215)

[Lighting System 21](#_Toc75105216)

[Flow Chart Diagrams 21](#_Toc75105217)

[Watering System 21](#_Toc75105218)

[Temperature System 22](#_Toc75105219)

[Lighting System 23](#_Toc75105220)

# Design

## Problem

In today’s dynamic life people tend to ignore the idea of home growing your own produce. Even though this is cheaper, tastier, and possibly safer for your health they prefer to buy fruits and vegetables from the shop without knowing what pesticides and grow methods have been used. The main reasons for this are the lack of time, lack of knowledge or confidence and lack of space to grow them yourself. The main aim of our product is to introduce a modern solution to these problems. With it you won’t have to worry for anything except choosing what crops you will want to cultivate.

## Our solution

Our Gardening Hub will be able to make decisions on its own, such as when to water the plant, turn on the grow light or add PH controlling solution to the water during the watering process. This will be achieved by using the predefined settings in the main controller in conjunction with independent sensors which gather data on a fixed time interval. Once all the data is in the controller, it will be processed and then the necessary action will be taken. By integrating a smart system in a gardening rig (no idea for better word) we can completely erase the above-mentioned problems. The only interaction the user has to do with the system is to put the seeds in the soil, add PH regulating solution to the reservoir around once every 6 months and collect the produce at the end of the growth cycle. The user will also be able to change the predefined settings or create a completely new growth cycle setting based on his desire. The system also allows for multiple hubs to be connected at the same time. This is where we introduce our distributed system plan. This is used when the systems use one main water tank to operate each private watering system. To evade the problem of emptying out the main water tank the systems communicate with each other, prioritizing which needs to be watered the most at the given moment. By doing this we ensure that the output volume won’t overpower the input volume of the water tank.

The product will provide to the entry level user a hassle-free environment to grow their first home grown vegetables and a modern way to experiment for the experienced user.

# Context Diagram

**Node-Red Dashboard**

Send:

Custom program

Commands

Receive:

Sensor values

Grow box IPs

Water levels

|  |
| --- |
| **Server -> Grow box** |
| Add node to network |
| Remove node from network |
| Assign program |
| Pause program |
| Resume program |

|  |
| --- |
| **Grow box -> Grow box** |
| Send water priority |

|  |
| --- |
| **Grow box -> Server** |
| Send climate data |
| Sensor error |

|  |
| --- |
| **Grow box -> Water tank** |
| Water request |

|  |
| --- |
| **Water tank -> Grow box** |
| Send water tank level |

|  |
| --- |
| **Dashboard -> Server** |
| Custom program |
| Commands |

|  |
| --- |
| **Water Tank -> Dashboard** |
| Water levels |

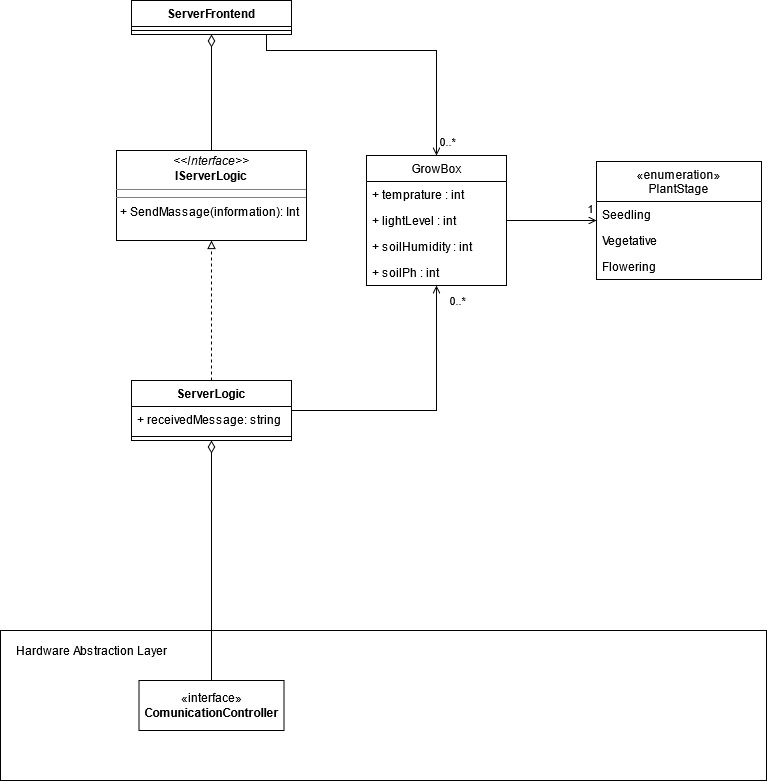
|  |
| --- |
| **Server -> Dashboard** |
| Growbox IPs |
| Sensor values |

# Class Diagram

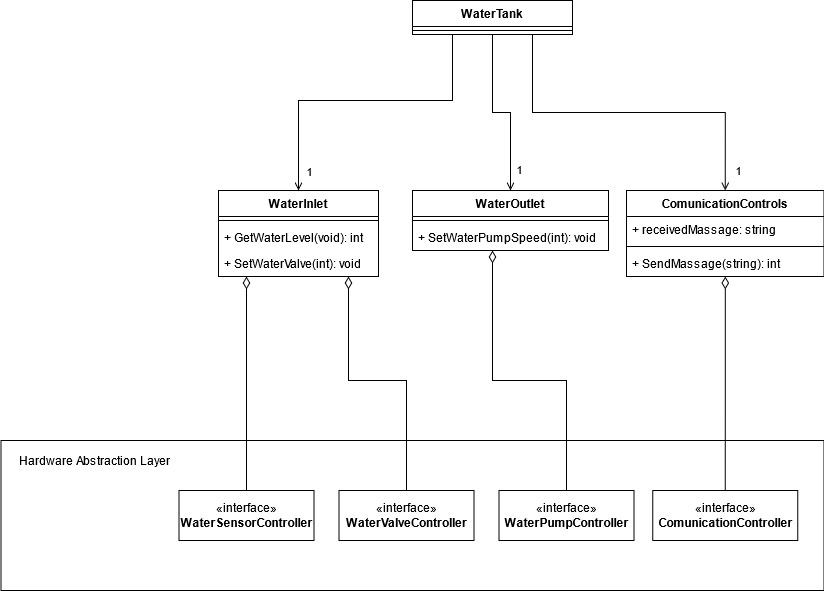
# Growbox:

# 

Server:



Watertank:



# Hardware

## Soil moisture sensor

Purpose: To monitor the soil’s moisture value

Reason: To provide the crop a good growing environment. The soil’s moisture value must be monitored, then the system is able to decide actions base on the reading.

Belonging system: Soil environment monitor system

## PH sensor

Purpose: To monitor the soil’s PH value

Reason: Similar as soil moisture sensor

Belonging system: Soil environment monitor system

## Water level detector

Purpose: To detect the water level in the water tank

Reason: To prevent water leak and not overfill the water tank, the water level detector is necessary. If no action is taken but the water level drops, then this could mean there is a leak happening. If the water level reached the upper limit of the water tank, then the system must stop the pump to prevent water overfill.

Belonging system: Watering system

## LDR sensor

Purpose: To detect the Light intensity outside the grow box

Reason: This sensor is used to monitor the environment light outside the grow box. Base on the value, the system will decide to turn off or on the grow light and by how much.

Belonging system: Lighting system

## Distance sensor

Purpose: To detect the distance between the top part of the crop and the top of the grow box

Reason: This sensor will be used to prevent the crop inside the grow box grows too high. Because the growing light could potentially burn the crop if the distance between them is too close.

Belonging system: Lighting system

## Heating

Purpose: Increase the temperature inside the grow box to match the desired temperature value.

Reason: To avoid low temperature which will cause the crop to die, a heater should be placed and turned on when the temperature reached the lower limited.

Belonging system: Air control system

## Water pump

Purpose: Pump water from the water source to the water tank

Reason: To transport water from the water source to water tank, a water pump is necessary.

Belonging system: Watering system

## Water valve

Purpose: Supply water to the crop in the grow box

Reason: To control the water flow, a water valve is necessary.

Belonging system: Watering system

## Fan

Purpose: Exchange air or decrease the temperature to match the desired temperature value

Reason: To avoid high temperature, and exchange the air, fan is necessary.

Belonging system: Air control system

## Grow light

Purpose: Supply the necessary light source for the crop in the grow box

Reason: To provide the crop necessary light source for growing, grow light is necessary.

Belonging system: Lighting system

## ESP32

Purpose: Provide WIFI and Bluetooth connection between the grow boxes.

Reason: This chip has both Bluetooth and WIFI functions. It will reduce cost compare with using Bluetooth and WIFI separately. The ESP32 itself is also very cheap too, and relatively reliable.

Belonging system: Communication system

## Water tubes

Purpose: Transport water from water source to the water tank, and from water tank to the grow boxes

Reason: Water tubes are necessary for transporting water.

Belonging system: Watering system

## Water tank

Purpose: Store water source

Reason: Water tank is necessary to store water.

Belonging system: Watering system

# Performance consideration

In this project, we have divided the actuators, sensors, and other components inside one grow box into separate subsystems, all of them will communicate with the main controller by wire communication to reduce the communication latency and instability.

The grow boxes will share a same water tank, the main control system will decide which grow box has highest priority to get water base on the sensor values.

Generally, each grow box will communicate with the server via WIFI, to store all the information gathered from sensors and actions took by actuators. For situations the WIFI communication is lost between one grow box and server, the grow box will use Bluetooth communication to find nearest grow box. This process will continue until the grow box which lost communication find the grow box which have the communication to the server. Then the information will be stored in the server among the grow boxes via Bluetooth and using WIFI finally.

We recommend user to put the grow boxes (if they buy multiple) closely. This will reduce the risk of lost communication and prevent possible leaks or water pressure drops because of the long-distance tubes.

# Communication protocol

### Start Char = ‘$ ‘

### Seperator Char = ‘: ’

### End Char = ‘# ‘

|  |  |
| --- | --- |
| **Name** | Add node to network (broadcast) |
| **messageID** | AddNode |
| **Message structure** | $AddNode:<IP># |
| **Parameters** | IP(of the new Grow box) |
| **Direction** | Server -> Grow box |
| **Usage** | Adding a new node to the network |
| **Acknowledgement** | Parameters: IP addr, <error>  &AddNode:<error>#  <error>  00 no error  01 invalid read |

|  |  |
| --- | --- |
| **Name** | Announce presence (unicast) |
| **MessageID** | Announce |
| **Message structure** | $rAnnounce:<IP># |
| **Parameters** | IP |
| **Direction** | Grow box -> Server |
| **Usage** | Alert server of presence |
| **Acknowledgement** | Parameters: <error>  &Announce:<error>#  <error>  00 no error  01 IP already exists  02 invalid read |

|  |  |
| --- | --- |
| **Name** | Remove node from network(broadcast) |
| **Command** | Remove |
| **Message structure** | $Remove:<IP># |
| **Parameters** | IP |
| **Direction** | Server -> Grow box |
| **Usage** | Removing a node from the network |
| **Acknowledgement** | Parameters: <error>  &Remove:<error>#  <error>  00 no error  01 IP doesn’t match |

|  |  |
| --- | --- |
| **Name** | Assign program(unicast) |
| **Command** | Program |
| **Parameters** | End time, temp, soil moisture, light intensity, light time cycle, Priority overwrite |
| **Message structure** | $Program:<endTime>:<Temp>:<soilMoisture>:<LightIntensity>:<lightCycle>:<Prio># |
| **Direction** | Server -> Grow box |
| **Usage** | Starting a new program on a grow box |
| **Acknowledgement** | Parameters: <error>  $Program:<error>#  <error>  00 no error  01 invalid read  02 node not registered |

|  |  |
| --- | --- |
| **Name** | Request program data (unicast) |
| **Command** | ReqProgSpec |
| **Parameters** | IP |
| **Message structure** | $ReqProgSpec:<IP># |
| **Direction** | Grow box -> Server |
| **Usage** | When Node is restarted request program specifications from node-red |
| **Acknowledgement** | Patamets: <error>  $ReqProgSpec:<endTime>:<Temp>:<soilMoisture>:<LightIntensity>:<lightCycle>:<Prio>#  $ReqProgSpec:<error>  <error>  00 node not registered  01 no program running  re-send program specs (also send: End time, temp, soil moisture, light intensity, light time cycle, Priority overwrite) |

|  |  |
| --- | --- |
| **Name** | Grow cycle completed (unicast) |
| **Command** | CycleCompleted |
| **Parameters** | IP, End time |
| **Message structure** | $CycleCompleted:<IP>:<EndTime># |
| **Direction** | Grow box -> Server |
| **Usage** | When the grow cycle is completed alert the user through the server dashboard |
| **Acknowledgement** | Patamets: <error>  $CycleCompleted:<error>#  <error>  00 no error  01 node not registered |

|  |  |
| --- | --- |
| **Name** | Send climate data (unicast) |
| **Command** | SendData |
| **Parameters** | IP, Temperature, Soil moisture, Light intensity, Humidity |
| **Message structure** | $Data:<IP>:<Humidity>:>:<LightIntensity>:<Temp>:<SoilMoisture># |
| **Direction** | Grow box -> Server |
| **Usage** | Periodically sending data from the plant box to the server to save |
| **Acknowledgement** | Parameters: <error>  $Data:<error>#  <error>  00 no error  01 node not registered |

|  |  |
| --- | --- |
| **Name** | Water level (unicast) |
| **Command** | WaterPrio |
| **Direction** | Grow box -> Grow box (all other grow boxes) |
| **Message structure** | $WaterPrio:<WaterLevel>:<Priority># |
| **Parameters** | Water Level, Priority |
| **Usage** | When the grow box needs water sends its needs to other grow box to spread out water usage |
| **Acknowledgement** | Parameters: <error>  $WaterPrio:<error>#  <error>  00 no error  01 node not registered |

|  |  |
| --- | --- |
| **Name** | Send water valve state (unicast) |
| **Command** | WaterValve |
| **Direction** | Grow box -> Watering system |
| **Message structure** | $WaterValve:<ValveState># |
| **Parameters** | Valve state |
| **Usage** | When the grow box needs water sends its needs to other grow box to spread out water usage |
| **Acknowledgement** | Parameter: <error>  $WaterValve:<error>#  <error>  00 no error, request accepted  01 request denied, insufficient water |

|  |  |
| --- | --- |
| **Name** | Pause program (unicast) |
| **Command** | Pause |
| **Parameters** | -- |
| **Direction** | Server -> Grow box |
| **Message structure** | $Pause# |
| **Usage** | Pausing currently running program |
| **Acknowledgement** | Parameters:  $Pause:<error>#  <error>:   1. Program paused, no error 2. No program assigned 3. Program already paused |

|  |  |
| --- | --- |
| **Name** | Resume program (unicast) |
| **Command** | Resume |
| **Parameters** | -- |
| **Message structure** | $Resume# |
| **Direction** | Server -> Grow box |
| **Usage** | Resuming paused program |
| **Acknowledgement** | Parameters:  $Resume:<error>#  <error>:   1. Program resumed, no error 2. No program assigned 3. Program already running |

|  |  |
| --- | --- |
| **Name** | Sensor error (unicast) |
| **Command** | SensorError |
| **Message structure** | $SensorError:<SensorType>:<errorType># |
| **Parameters** | <error>, Sensor type, error type  Error types:   1. Invalid reading 2. No response |
| **Direction** | Grow box -> Server |
| **Usage** | A sensor in the grow box is not working |
| **Acknowledgement** | Parameters: <error>  $SensorError:<error>#  <error>  00 no error  01 node not registered |

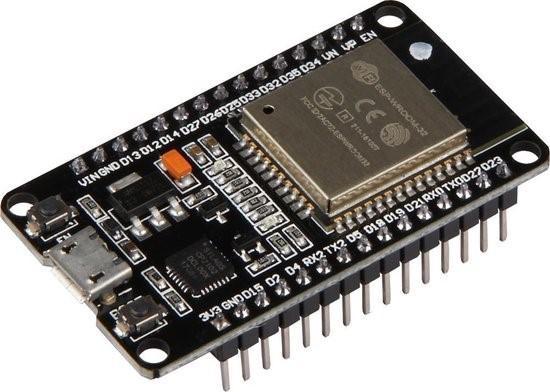
|  |  |
| --- | --- |
| **Name** | Water leak (unicast) |
| **Command** | WaterLeak |
| **Message structure** | $WaterLeak# |
| **Parameters** | -- |
| **Direction** | Watering system -> Server |
| **Usage** | When the watering system detects a leak |
| **Acknowledgement** | Parameters: <error>  $WaterLeak:<error>#  <error>  00 no error  01 node not registered |

|  |  |
| --- | --- |
| **Name** | WaterTank announce |
| **Command** | WaterTankAnnounce |
| **Message structure** | $WaterTankAnnounce:<IP># |
| **Parameters** | IP |
| **Direction** | Watering system -> Server |
| **Usage** | Alert server of presence of water tank |
| **Acknowledgement** | Parameters: <error>  $WaterTankAnnounce:<error>#  <error>  00 no error  01 IP already exists  02 invalid read |

|  |  |
| --- | --- |
| **Name** | WaterTank send data |
| **Command** | WaterTankData |
| **Message structure** | $WaterTankData:<waterLevel># |
| **Parameters** | waterLevel(%) |
| **Direction** | Watering system -> Server |
| **Usage** | Sends the server the current waterLevel of the waterTank |
| **Acknowledgement** | Parameters: <error>  $WaterTankData:<error>#  <error>  00 no error  01 invalid read |

# Communication protocol description

Each grow box will be powered by an ESP32 board which is capable of Wi-Fi and Bluetooth communication. The grow boxes will be able to communicate with each other via Wi-Fi to establish priority for water and electricity. The boxes will also communicate with another ESP32 that hosts the main server where all of the data for each plant is stored as backup in case of any power outages or other problems.



The ESP32 that hosts the server will also be connected to the main UI in which the user can add/remove plants and edit programs or make new ones. In case of a malfunction of one of the modules the board will switch to Bluetooth and send its data to the nearest box to back up its data and communicate with each other that way.

## **Rich Shield Sensors**

Graphical user interface

Description automatically generated

Humidity sensor – to measure the humidity in the box we will be using the DHT11 sensor which uses I2C communication, this data will be used to regulate appropriate air humidity for the plant..

Temperature sensor – to measure the temperature in the box we will be using the NTC sensor which provides data via an analog pin, this data will be used to regulate appropriate temperature for the plant.

LDR sensor – The LDR sensor will send its readings to the main controller via an analog pin and we will use the data to regulate the lightning in the box.

A close-up of a stethoscope

Description automatically generated with low confidenceSoil humidity sensor – The soil humidity sensor will send its readings to the main controller which based on the data it receives will send requests to the watering tank. This sensor communicates via analog signal.

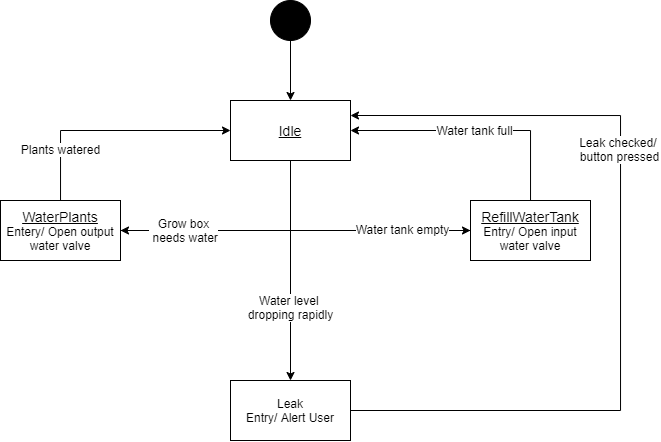
Watering system – The watering system is going to be powered by an ESP32 as well and it will control the valves based on messages it receives from other boxes. The box that is sending the message will be decided by the boxes themselves based on which one needs to be watered the most and it will send its message to the controller controlling the valves which will open the required valve.

Heating system – For the heating system we are going to be using an LED panel which will be controlled by an algorithm which decides when the plants need light and heating. If the lighting system should be turned on, the ESP32 powering the grow box will receive a command from the main controller letting it know the lighting should be turned on.

# State Diagrams

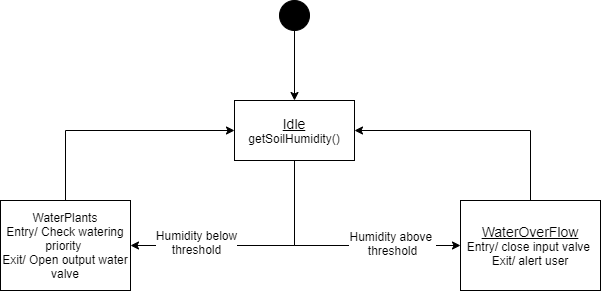
## Watering system (water tank)

|  |  |
| --- | --- |
| Message | Description |
| Water tank empty / full | Water level sensor inside water tank will provide us with the levels of the water tank. |
| Growbox needs water / plants watered | Soil humidity sensor will keep us up to date to when the plant will need watering. |
| Water levels dropping rapidly | While no action is happening, if the water level sensor in the tank detects the levels dropping it should alert the user for a leak. |
| Leak checked / button pressed | The device will leave the state Leak once it is checked, otherwise it will keep alerting the user until checked. |



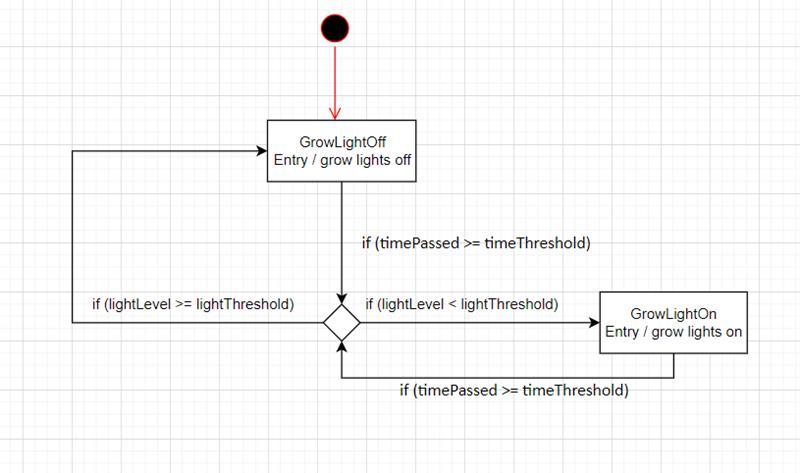
## Watering system (Grow box)

|  |  |
| --- | --- |
| Message | Description |
| getSoilHumidity() | This function is meant to be ran on loop, it will obtain the soil humidity from the sensor. |
| Close / Open water valve | The water output will be controlled by a selonide valve. |
| Check watering priority | Before watering, the grow box will check if other plants need water. If they do, it will wait until it is its turn to water the plants. |
| Humidity above/below threshold | The humidity in the soil will let us know when the plant needs watering. |



## Lighting System

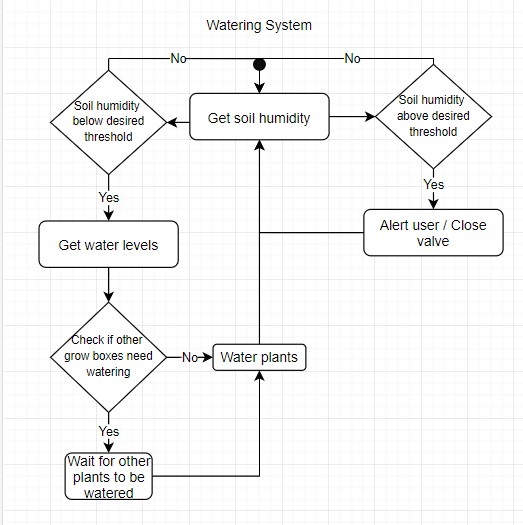
|  |  |
| --- | --- |
| Message | Description |
| timePassed >= timeThreshold | The plant will have a light schedule depending on what stage it is. Ex: 12hr on / 12hr off |
| lightLevel >= lightThreshold | The light will be put to a certain strength depending on what stage it is. Ex: x lumens for seedling stage |



# Flow Chart Diagrams

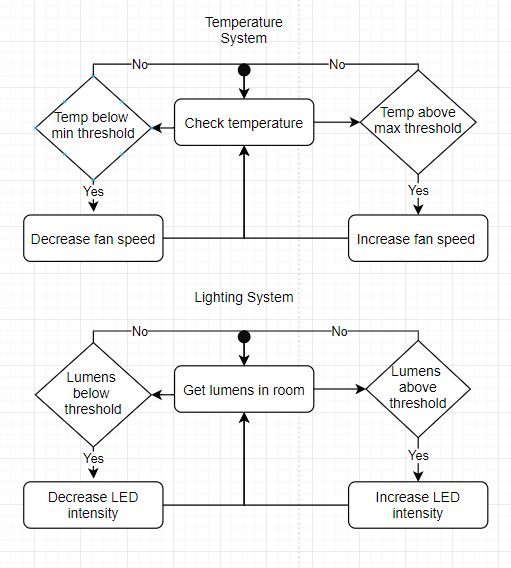
## Watering System

The following flow chart will describe the flow of the watering system within the grow box, it does not include the water tank. The device will constantly get the soil humidity and if the soild humidity is above a desired threshold it should alert the user that his/her plant is in danger. Otherwise, the system will get the water levels, from the water tank, and check with other grow boxes to see water priority. After knowing water priority of all connected growboxes, if there is no other plant that needs water urgently it will be watered.



## Temperature System

The following flowchart will describe the flow of the temperature system. There will be a fan inside of each grow box, if the temperature does not meet with the desired temperature the pwm signal will be adjusted until the requirements are met.



## Lighting System

The following flowchart will describe the flow of the lighting system. The light is supposed to be adjustable to a certain level and the on / off is controlled by a time variable. That being said, I do not think the on / off factor should be included in the following diagram. The user will be able to choose an intesity for the LED. If the lumens do not match, increase/decrease slowly until requirements are met.

