SYSC 4907
Final Year Report

Automated IoT Hydroponics System

Date: 2017-2018

Joshua Armel -- 100889933

Haris Ghauri -- 100910613

Supervisor: Professor James Green

Department of Systems and Computer Engineering
Faculty of Engineering
Carleton University

Table of Contents

Abstract

The ability to access fresh local produce year round is a problem in many areas of the world. Whether this is because of climate, location or pesticide use. In response to this, people have been exploring growing their own produce at home and indoors. This project is designed with the average consumer in mind, and is scaled to fit their needs. This automated hydroponics system will allow people to produce their own fresh produce in the comfort of their own home, with very little input from their end. This system is targeted at those consumers who wish to have their own local produce all year round in any climate, and whose agricultural skills are limited. This also fits the need for people who do not have the time or effort to have a traditional home herb garden. This project was developed in two main parallel parts: the embedded system and physical hydroponics setup, and web based UI and image processing. This allowed us to specialize during our development cycles. We have successfully integrated both parts of the project together, and can communicate between them in a robust and easy to use manner. The system has performed as expected over the last few months, and is giving us useful growth and disease detection data which is used to verify that the system is working correctly.

List of Figures

List of Tables

1. Introduction

Hydroponics is a technique used to grow plants without the use of soil. This requires the use of nutrients dissolved in a solution that passes over the roots of the crops being grown. This method of agriculture has many advantages including water usage efficiency(usage is less than 8% of conventional production methods: [1]), greater yields [1], and can provide local access to fresh produce.

1.1 Problem Background

Currently, there is a growing number of people who are looking to purchase their produce locally and organically. They understand the harmful effects of shipping in produce from around the world, in terms of transportation cost and freshness of the end product. There is also a concern regarding the use of Genetically Modified Organisms (GMO), which makes crop pesticide resistant, thus enabling large quantities of these toxins to be spray onto them. Additionally, there is also a concern on how to monitor the progress of growth and the health of plants. A typical consumer does not want to visit his/her locally grown plants in order to observe the parameters, and even if he could visit, he would not be able to understand those details properly.

1.2 Problem Motivation

An automated or semi-automated home hydroponics system would address that demographic and solve this dilemma. It would allow the consumer to monitor his/her plants directly through a web/mobile interface. The agricultural benefits of this project could be expanded into a more commercial setting, with the automation of large scale hydroponic systems. This agricultural technique would be advantageous as there are huge water savings to grow produce this way (upwards of 94%[1]). With the reduced water needs, dry geographical regions and regions that suffer from drought would greatly benefit from adopting this growing style. In the commercial setting of a greenhouse, this system could be distributed to remote communities that do not have access to fresh produce such as ones found in Nunavut.[2]

1.3 Problem Statement

The main three problems this project addresses are, being accessible to users with no growing experience, supply edible produce in a safe and efficient manner, as well as tracking the plant growth and health during the crops life cycle.

1.4 Proposed Solution

Currently, there are a couple hydroponics projects involving machine learning that are being studied in the industry. One from IKEA [4], and one hobbyist project found on instructables.com[5]. Our project differs from these, as we will be adding elements of image processing to our hydroponics system. Periodic images will be taken of our plants, and the health of the plants (nutrient deficiencies and/or diseases) will be determined. Figure 1 illustrates our proposed solution.

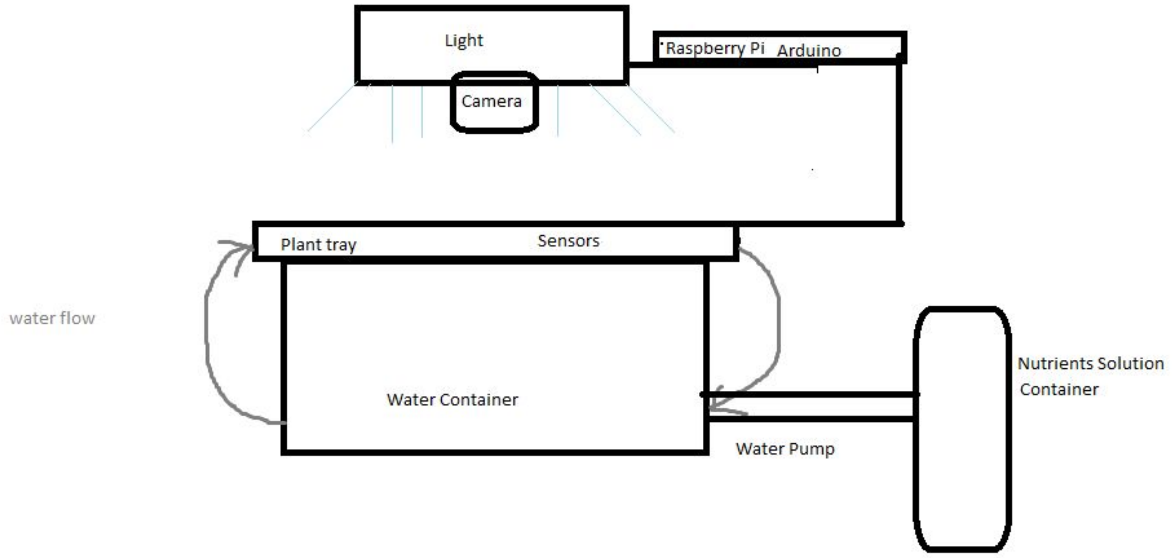


Figure 1: Rough Sketch

1.5 Functional Requirements

- The System should be able to sense the intensity of light around it. Based on that, it should automatically adjust its own artificial light source.
- The System should be able to monitor the temperature & humidity, and report this data to the web server (this includes the water temperature in the aquarium).
- The system should be able to detect the pH level, and the dissolved solids in the water (nutrient level approximation by using a ppm or micro siemens meter).

- All system parameters must be monitored and controlled from a web interface/smartphone application.
- A camera will be attached to the system and provide a live feed (or frequent still images) to the web interface which can be accessed via the smartphone application.
- All the information from the sensors will be recorded in a database to make the future crop cycle more efficient (higher yields).
- The yields will be determined by weighing the product after each harvest.
- The system should be able to generate an email or notification in case of a temperature/humidity issue, nutrient dosing error, and other urgent cases.
- The creation of different profiles for individual plant species which will take into consideration the optimal temperature range, light/darkness cycle, nutrient requirements as well as the plant health tracking such as potential deficiencies.
- Health and growth of plants should be tracked through image processing.
- The lighting system will be directly above the grow bed, simulating the sun at high noon, thus no rotation or the lights/bed will be required.

1.6 Overview of the Remainder of Report

The report follows a traditional Engineering report format. Chapter 1 described the problem and the proposed solution. Chapter 2 talks about the ‘Engineering’ involved in this project. This consists of Engineering Professionalism, Project Management and Health and Safety. Chapter 3 is the heart of this report where background and terminology is described followed by the architecture of the systems, the process flows, the methods used and the results obtained. The last chapter 4 contains the conclusion, possible future work and recommendations. Lastly Chapter 5 contains the references.

2. The Engineering Project

2.1 Health and Safety

Due to the nature of this project, there are some electrical safety concerns that were considered during construction and use of the project. Since we are using relays to control AC current, and we have flowing water in our system, there are real concerns for electrical shock and short circuiting the system. Another thing we needed to consider was the artificial lighting, as it would be visible to anyone who is around the system.

2.1.1 Alternating Current and Electrical Shock

To minimize the risk of electrical shock due to the 120V 15A current, the Ontario Electrical Safety Code[7] guidelines were followed regarding plastic electrical boxes. The relays which control the AC circuit are contained inside the box, and no exposed wires are visible, thus preventing contact with any 120V lines. The safety code requires that metal boxes be grounded, while plastic ones don't. Since all the wiring was done by hand, we made out electrical box grounded regardless.

2.1.2 Light intensity of red and blue spectrum LEDs

Upon research into using LED grow panels which focus on the red and blue spectrum, it became apparent that there was an associated health risk for those viewing the light source. Reference [8] describes how we are sensitive to the blue spectra, and how this light can be damaging to our eyes. Reference [9] discusses how our eyes are not as sensitive to the intensity of the blue spectra compared to others. With both of these in mind, it can be concluded that having high intensity light the primarily focuses on the blue spectra for photosynthesis, will not be registered the same by our eyes as white light would. This could lead to damage to the retina if the light source was observed for any considerable amount of time. To protect ourselves and others around us, we wrapped the system using Mylar film. This film is reflective and will prevent a majority of the light from escaping, while increasing the overall light output to the plants.

2.2 Engineering Professionalism

During the development of our project, we incorporated professional practice aspects. We kept good communication between members of the group, and would meet in person at least once a week. We kept in contact throughout the week using google docs and FaceBook messenger to update each other on our progress. Nearing the end of our project, we met up on both weekend days in order to expedite our progress.

Punctuality is another aspect of professionalism that was adhered to while working on our project. Our team members would always meet on time for our weekly team meetings, presentations and poster fair. If there were occasions in which we were unable to make it on time, notice would be given to the other group members ahead of time, which is consistent with professional work behaviour.

During decision conflicts, we remained calm and respectful and listened to the other parties concerns. Compromises were made in order to achieve positive outcomes for both parties. These design decisions are outlined in section 3.3 of this report.

2.3 Project Management

We managed our project through Google Docs by updating our project proposal with our new design decisions as well as tasks for the team members to do. In terms of task priority, during our weekly meetings, our weekly goals were determined, as well as updating our backlog of tasks to do. This was recorded on paper in notebooks while meeting with our supervisor.

Our code was synced through a public GitHub repository, in order to include version control software which allowed us to view each other's code and become familiar with the technologies used.

2.4 Justification of Suitability for Degree Program

For this project, the technical objectives are discovered by recognising the functional and business requirements. These requirements will be gathered from farmers' needs. Since the project is driven by its requirements, it gives us the opportunity to act as DevOps. **DevOps** is a software engineering practice that aims at unifying software development (Dev) and business operation (Ops). The technical objectives are further divided into small tasks, each of which are either tested or implemented by both members of the team. Related tasks are grouped together to form a subprojects, increasing modularity and generality.

2.5 Individual Contribution

2.5.1 Project Contribution

Work	Member
Physical Hydroponic Setup	Joshua
Web App and Raspberry Pi Web Associated Code	Haris
Serial Communication and I/O Devices using Arduino	Joshua
Image Processing	Haris

Table 1: Project Contribution

Joshua Armel worked on the construction of the hydroponics model, including the tank setup and growing the starter plants. The Arduino code controlling the actuators and the sensors and calibration of the sensors. The serial port communication between the Raspberry Pi and the Arduino as well as the health and safety features mentioned in section 2.1 were also completed.

Haris worked primarily on the client side web app and the image processing code associated with it. Haris also wrote python scripts that read from firebase database and sends commands to Arduino and vice versa.

2.5.2 Report Contribution

Report Sections	**Contributor**
Abstract	Joshua
Section 1.0 - 1.3	Joshua
Section 1.4 - 1.5	Haris and Joshua
Section 1.6	Haris
Section 2.0 - 2.2	Joshua
Section 2.3 - 2.4	Haris and Joshua
Section 2.5	Haris and Joshua
Section 3.1.1	Haris
Section 3.1.2	Joshua
Section 3.1.3	Haris
Section 3.2.1	Joshua
Section 3.2.2 - 3.2.4	Haris
Section 3.2.5	Joshua
Section 3.2.6	Haris
Section 3.3.1	Joshua
Section 3.3.2 - 3.3.6	Haris
Section 3.3.7	Joshua
Section 3.4.1 - 3.4.3	Haris
Section 3.4.4 - 3.4.5	Joshua
Section 4.1	Joshua
Section 4.2	Haris

Table 2: Report Contribution

3. Technical Sections

3.1 Background and Terminology

The following sections go over the background and terminology needed to understand the actual design choices, the system architecture and different strategies to solve the problem. It is important to understand the system process flow at a very high level before jumping into the terminology. Hence, a rough sketch of the system is placed right here in Figure 2 with their process flows. Process flow and system architecture is described in much detail in section 3.2 and 3.3.

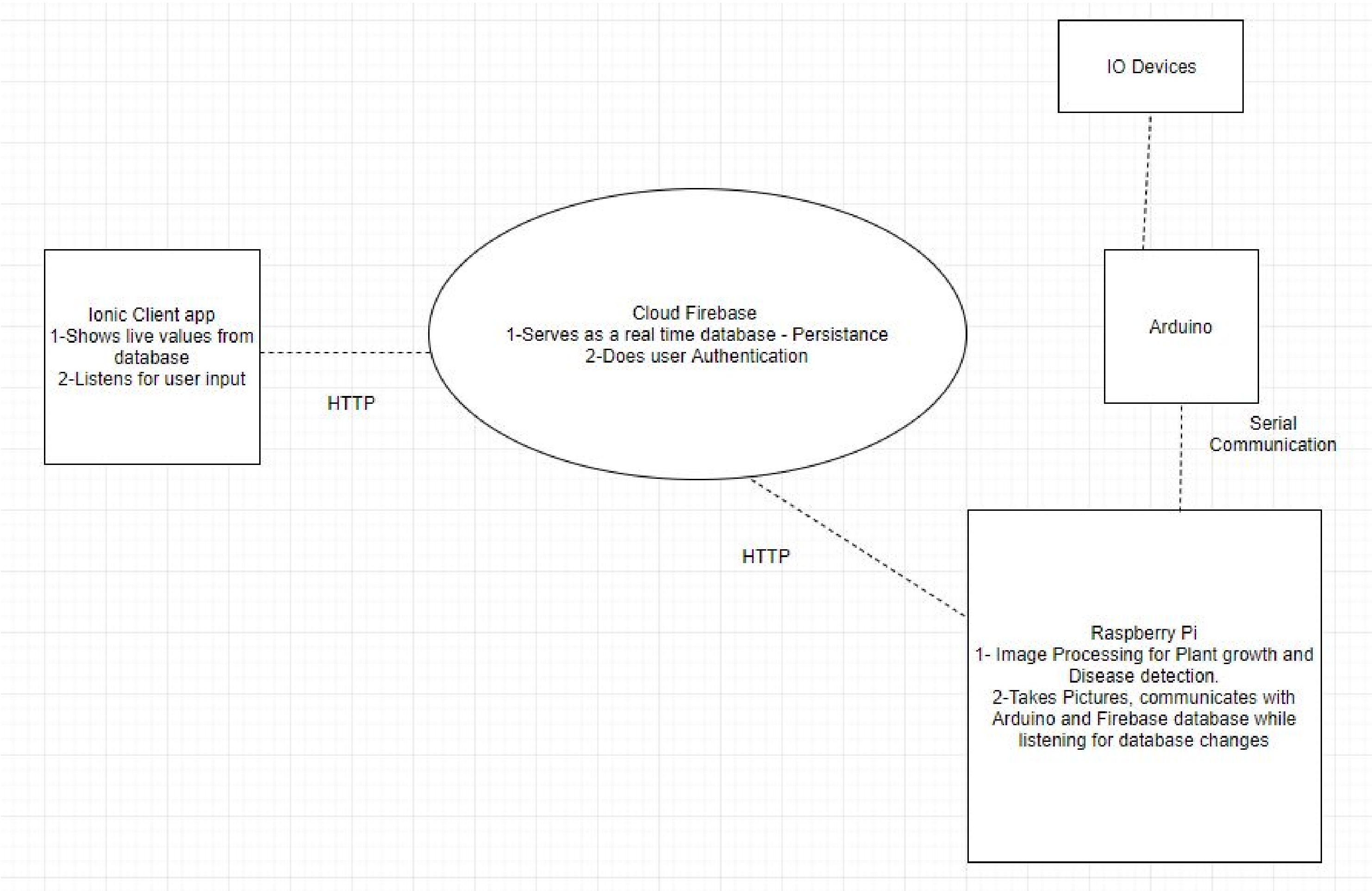


Figure 2: Rough Sketch of System Architecture

As it can be observed, there are different components of the system ; 1) Ionic Client App, 2) Cloud Firebase where data persists and authentication takes place, 3) Raspberry Pi which communicates with the two ends (Arduino and Firebase Cloud) and also does image processing, and 4) Arduino which controls the IO devices and receives command/sends updates to Raspberry Pi. With this process flow in mind, it would be easier for the reader to have the motivation to read the following terminologies.

3.1.1 Image Processing Concepts

Image Processing is done in our project to detect plant growth and plant diseases. Prior to understanding how image processing is used in our project, some concepts on image processing must be understood.

An image is nothing but a 2-dimensional (X,Y) or a 3-dimensional (X,Y,Z) array data structure.

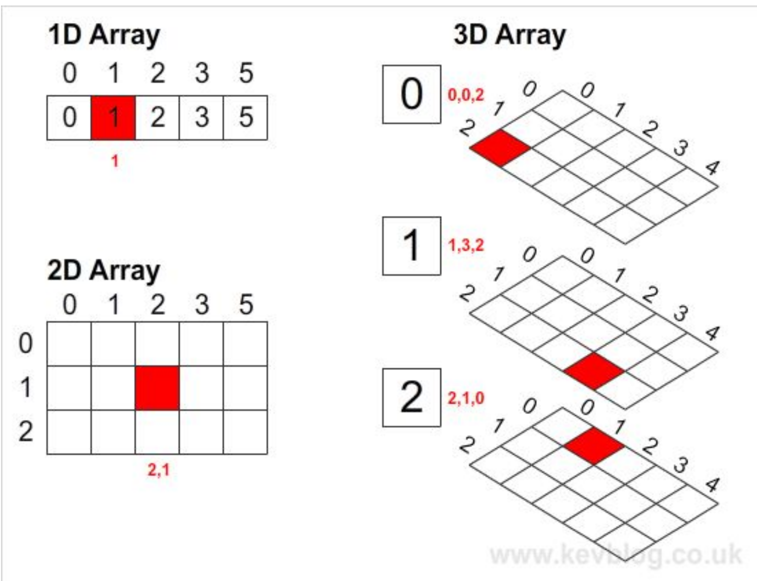


Figure 3: 2d vs 3d Array[10]

The values X and Y are very easy to understand. They represent the dimension of an image, i.e. the height and the width. However, the value Z varies, depending on how the image is being stored. The value of Z is 3 in most cases.

An individual pixel is represented by a location in X-Y plane with coordinate (x,y).

3.1.1.1 Binary Image

In the case of a binary image, the array is 2-dimensional with possible values of either 0 or 1 at any individual (x,y) location on the X-Y plane. This will result in a pixel which is either black or white but nothing in between.

3.1.1.2 Gray Scale Image

In the case of a gray scale image, the array is 2-dimensional as well. However, the values could be anything between the range 0-255. This will result in a pixel which could represent a range of colour between black and white as shown in the figure below:

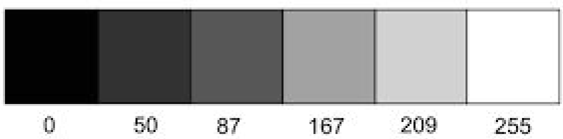


Figure 4: Grayscale Pixels [11]

3.1.1.3 RGB Image

In the case of an RGB image, the array is 3-dimensional with Z = 3. This allows 3 different values to be stored at any arbitrary coordinate (x,y) instead of just 1. These 3 values represent 3 distinct colors : Red, Green and Blue. The possible values for these colors can range anywhere between 0-255 which basically tells how much of each colour is present on this specific pixel. It's the combination of these 3 colours which enable us to see all the different colours in the visible spectrum.

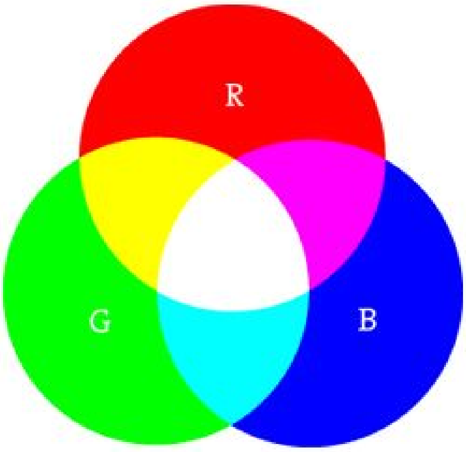


Figure 5: RGB colour venn diagram. [12]

3.1.1.4 HSV Image

In the case of HSV image, the array is 3-dimensional as well with Z=3. Just like an RGB image, this allows 3 different values to be stored at any arbitrary coordinate (x,y). However, these 3 different values represent the colours a little differently. The 3 values represent Hue, Saturation and Value. In simple words, Hue represents the actual colour, Saturation represent the amount of white in Hue and Value represent the intensity of the colour itself. The lower the intensity means that the colour is more similar to the black. The higher intensity means that the colour is more similar to itself.

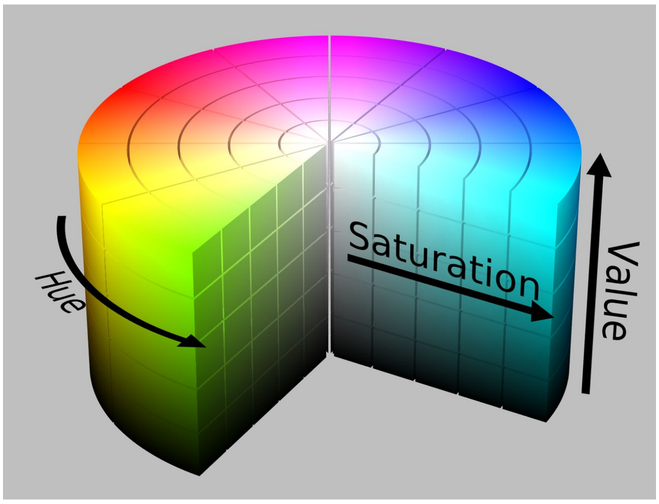


Figure 6: Hue Saturation Value colour diagram. [13]

3.1.1.5 Summary of Different Image Models

Image Model	Dimensions	Possible values in a Pixel (x,y)	Possible Colour
Binary Image	2-Dimensional	0 or 1	Black, White
Gray Scale Image	2-Dimensional	0-255	Black - Gray - White
RGB Image	3-Dimensional	(0-255,0-255,0-255)	Visible Spectrum
HSV Image	3-Dimensional	(0-255,0-255,0-255)	Visible Spectrum

Table 3: Summary of Different Image Models

3.1.1.6 Kernel Convolution

Using a process called kernel convolution, many manipulations can be done on the image. A kernel of size n * n is passed over the image of size M * N where n is of a much lesser value than M or N. A kernel is passed over each pixel and is replaced by the product of the kernel values of the surrounding values of the pixel. This technique is used for various applications namely: Blurring, Sharpening or Edge Detection.

3.1.2 Physical Components

3.1.2.1 Contactor Relays

Contactor relays are essentially electromagnetic switches that are controlled by a small current, relative to the current going through the switch. This is useful as to us to control our AC side since the relays can handle high current applications. In the following diagram you can see the electromagnetic coil, which will control the flow of current between the normally open (NO) contact and the normally closed (NC) contact.

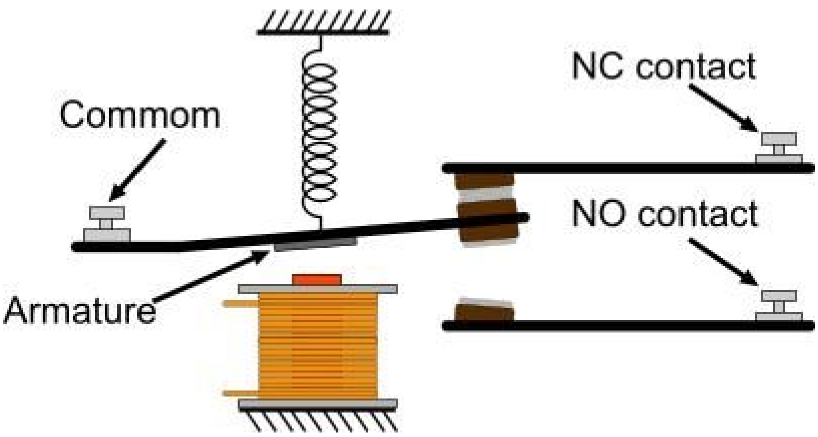


Figure 7: Electromagnetic relay operation diagram.

3.1.2.2 Data Through Analog Ports

The analog ports on the arduino are necessary in order to read in data from the electric conductivity sensor (EC) and the pH sensor. The analog ports differ from the digital ports in one main aspect. Digital ports can only handle binary values of either high or low (1 or 0). This is not useful for sensors, whose values can be a much larger spread than two values. The analog port handles this, as it reads in a value which is mapped to a voltage between 0v and 5v in increments of 4.9mV. This allows for a potential 1,020 unique values to represent EC and pH values of the system within the sensors given ranges. This is adequate for our purposes.

3.1.2.3 Arduino Microcontroller

The Arduino Uno consists of a microprocessor and controller. The board also contains a universal serial bus (USB), 6 analog ports, and 14 digital ports, 6 of which are pulse width modulation (PWM) ports. This microcontroller is programmed with a mix of C and C++ features combined into their proprietary language. This board has built in RAM and Flash memory at 2KB and 32KB respectively. This will provide enough RAM and far more than adequate amount of flash memory to store the code and operate the sensors in real time.

Figure 8: Arduino Uno that is used in our project.

3.1.2.4 Raspberry Pi Microcomputer

The Raspberry Pi is fundamentally different to the Arduino. The Pi in addition to the components that the Arduino has, consists of upgraded hardware, Linux operating system, and a video controller that can display the user interface and operating system. These added components makes the Pi a small computer, which is capable of doing all the processing a normal computer could, like connecting to the internet as well as controlling and programming microcontrollers like the Arduino. The lack of analog ports on the Raspberry Pi and the small number of I/O ports would not suffice for our needs, and therefore we needed to pair this with an Arduino microcontroller.

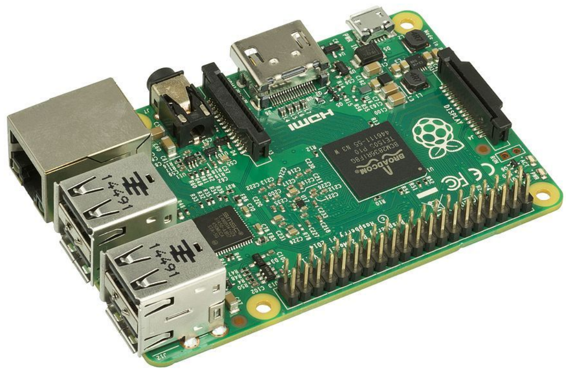


Figure 9: Raspberry Pi Model B used in our project.

3.1.2.5 Serial Port Communication

The Raspberry Pi and the Arduino microcontroller are connected via their universal serial bus (USB) ports. This port is used by us a way to transmit messages between the two devices. The messages are sent as strings and then encoded as bytes to be transferred over the wire. The reverse happens on the receiving end as the bytes are decoded to a string representation. When sending messages between devices, we used a simple convention and protocol to make the transfer easy and robust. When sending a message, the string is prepended with the name of the sender (PI or Arduino), so the devices always know who sent the message. The sending device will also wait 3 seconds for an acknowledgement message to arrive, or it will assume the transmission has failed. There is no retransmission attempt, however this is logged for later use, and the user is notified of the failure. (possibly notify user by email or UI message for Pi side, led for arduino side).

3.1.3 Software Decisions and Framework

3.1.3.1 Persistence -- NoSql Vs Sql

Sql databases have a predefined schema while NoSql databases have a dynamic schema which means we can add data without defining their structure. This makes NoSql superior but it does add overhead validating costs.

3.1.3.2 Client Side Application - Ionic

Ionic is an open source application development platform for hybrid development. It allows the creation of web apps, Ios apps and android apps. It is based on another front end web applications framework called Angular. Ionic uses Angular architecture and adds additional user interface functionalities to it. Ionic uses another technology called Apache Cordova to convert code written for web platform to native IOS and/or Android devices. The following diagram better illustrates the relation between these technologies.

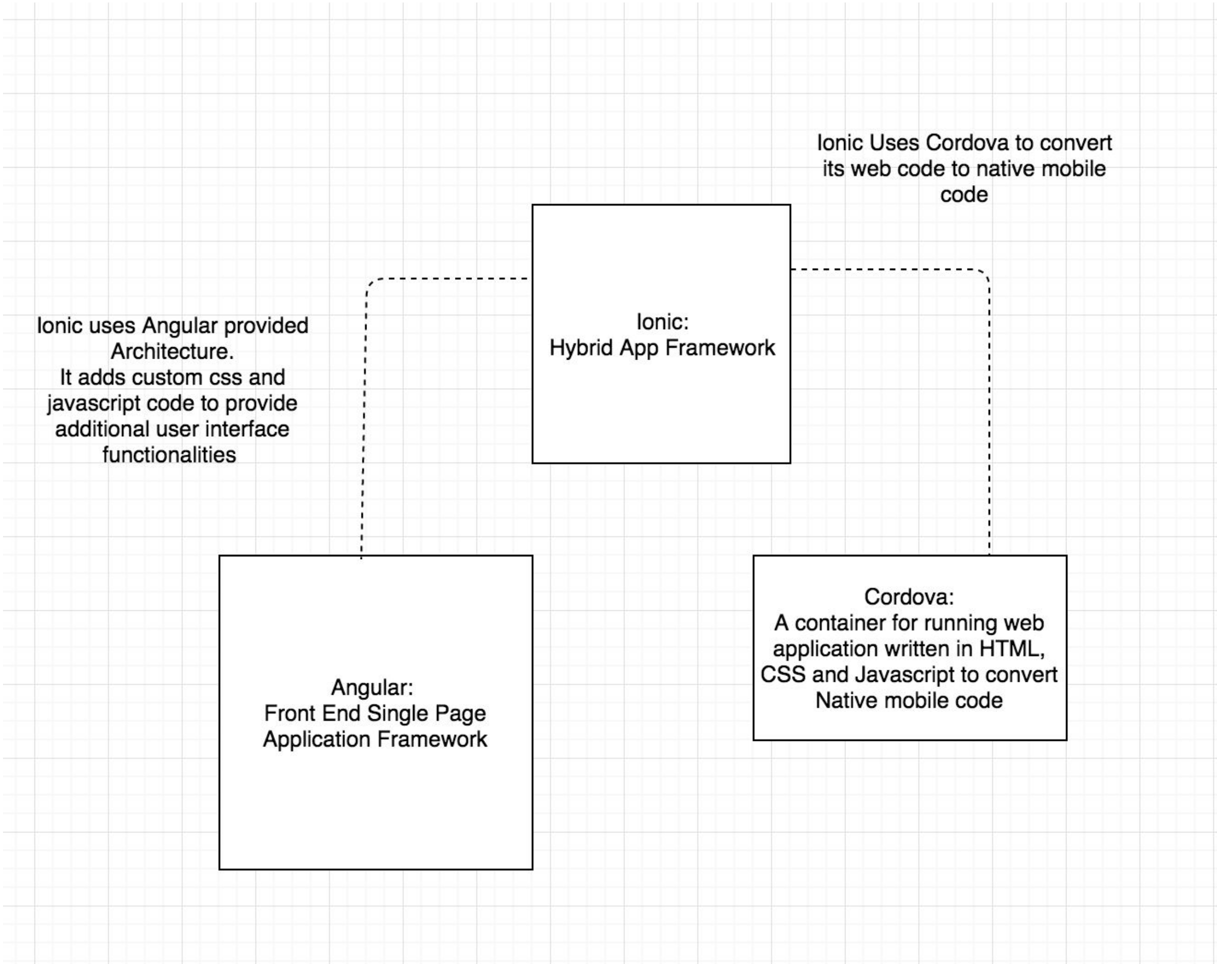


Figure 10: Ionic Framework Core

3.2 System Architecture, Design and Process Flows

3.2.1 Physical Setup and Space Accommodation

This project requires some Space, approximately 19 cubic feet = 2.5 feet x 2.5 feet x 3 feet. Canal Building 6107 has a desk which can be used to keep the project container and other miscellaneous components. Also the room has a sink which can be used to fill up the water container.

Figure 11: Core hydroponic system structure.

3.2.2 Initial High Level Architecture

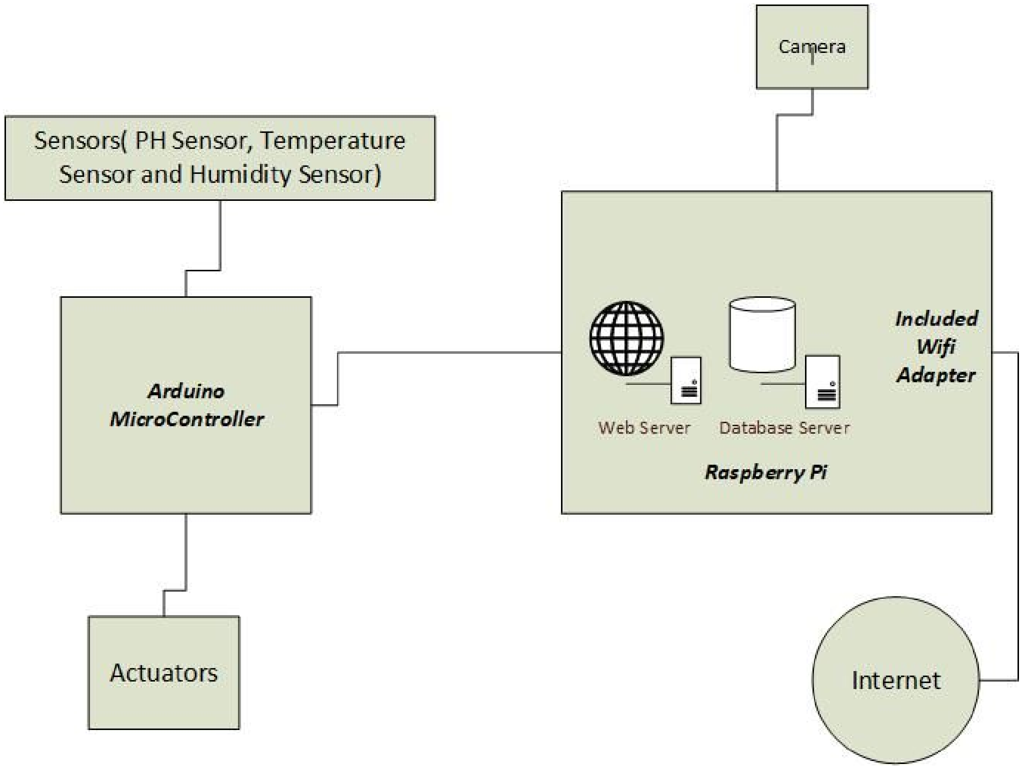


Figure 12: Initial High level Architecture.

The above figure represents the initial architecture of the System. Raspberry Pi because of its general computing capabilities was supposed to perform multiple programs which are as follows:

Web Server:

-Host Web Application that reads and updates database

-A thread that gets triggered each time a user performs actions on the web app and writes it on the serial port.

Program Listening on Serial Port:

-A program that reads sensor values from the serial port

Image Processing Program:

-A program that takes images at certain intervals, does image processing and updates the database accordingly.

3.2.2 Problems With the Initial Architecture

Since, Raspberry Pi was placed in a University Network and the University IT administration do not allow port forwarding on University Routers, therefore a new architecture design had to be proposed. This new architecture eliminated the need for the Raspberry Pi to act as a web server, rather, it was pushing data onto a central firebase database. This data can be viewed by client sided application by authenticated users.

More reasons for choosing Firebase Service(s) are discussed in Section 3.3.2.

3.2.3 Revised High Level Architecture

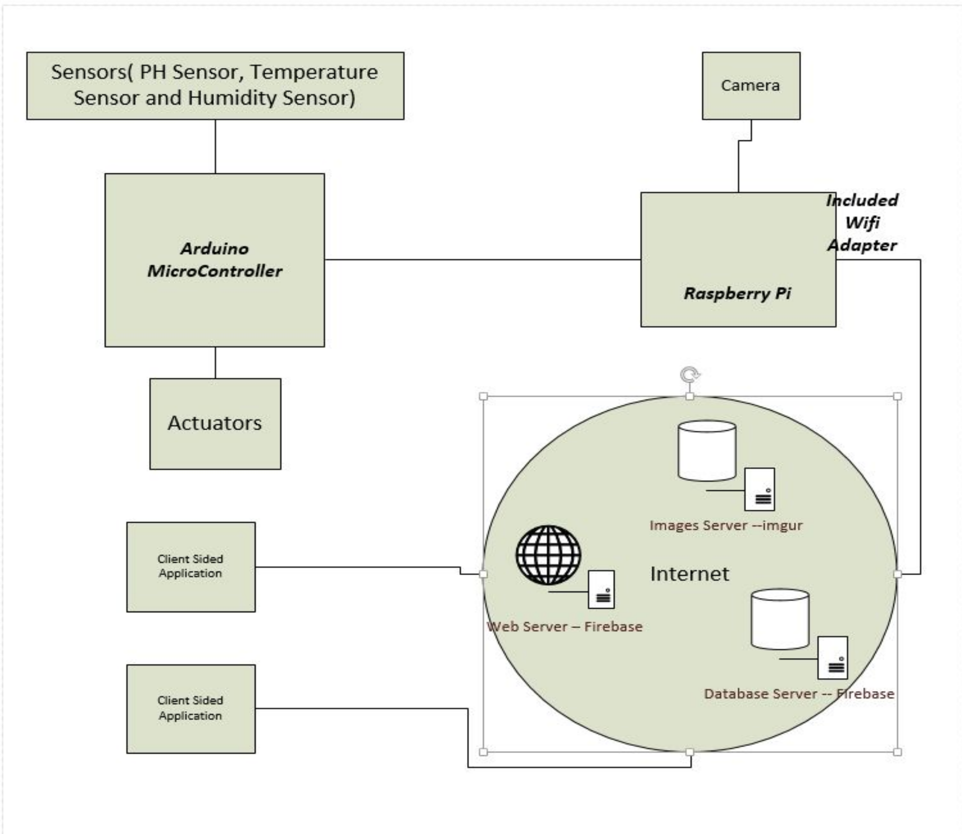


Figure 13: Revised High Level Architecture.

In this new Architecture, the Raspberry Pi handles two tasks instead of three.

Program Reading and Writing on Serial Port:

-reads sensor values from the serial port.

-listens for changes on firebase database server, writes those changes on the serial port.

Image Processing Program:

-A program that takes images at certain intervals, does image processing and updates the database accordingly.

3.2.4 Process Flow

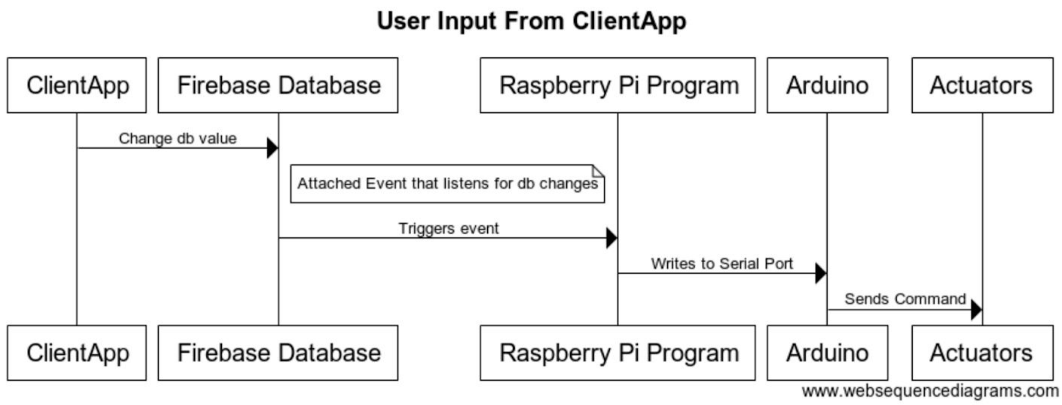


Figure 14: User Input Sequence Diagram.

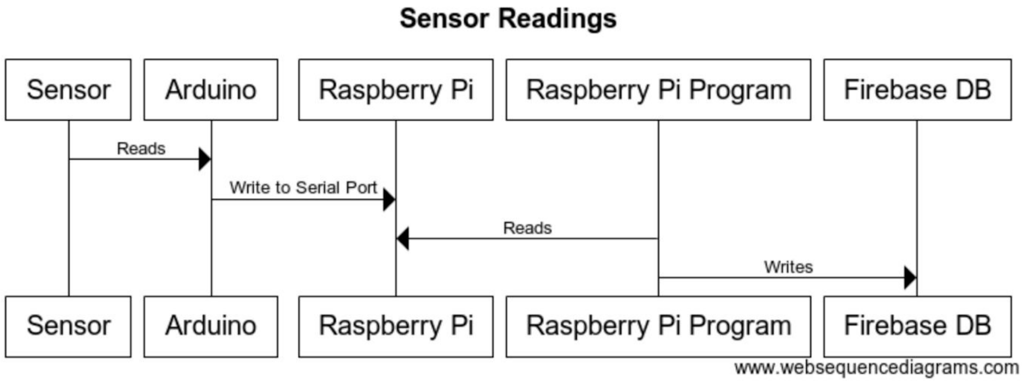


Figure 15: Sensor Reading Sequence Diagram.

3.2.5 Circuit Architecture

For the wiring between the microcontrollers and their I/O connections, common 18 gauge electronics wire was used, as the amperage will not be above 40mA. There is no safety concern with using this type of wiring for devices that draw less than 1 Ampere. There is a safety concern when we deal with the relays however, as these are required to be connected to 120V 10A-15A AC cables. In doing so safely, and in following safety codes, some design decisions needed to be made. For example, all the exposed electrical wiring needed to be contained and properly insulated to eliminate the risk of electrical shock. This is done by using an electrical box. There are two choices for electrical boxes, metal and plastic. When investigating our options, it was discovered that if using a metal electrical box, we would need to have this grounded to the circuits main line or else it would conduct. Since a plastic box does not have this issue as it is made of an insulating material, this is our ideal choice. The relays and the outlets are contained inside the plastic box. The cable choice was also taken under careful consideration as to fall under the correct safety codes. For the wiring, 12 gauge was chosen as it is the standard for electrical box configuration. This allows us to pull a maximum of 41 Amperes safely through the cable. This is overkill as the circuit could never draw more than 20 amperes through the wall. This extra buffer region will however allow much higher operating temperatures as well as low air circulation areas.

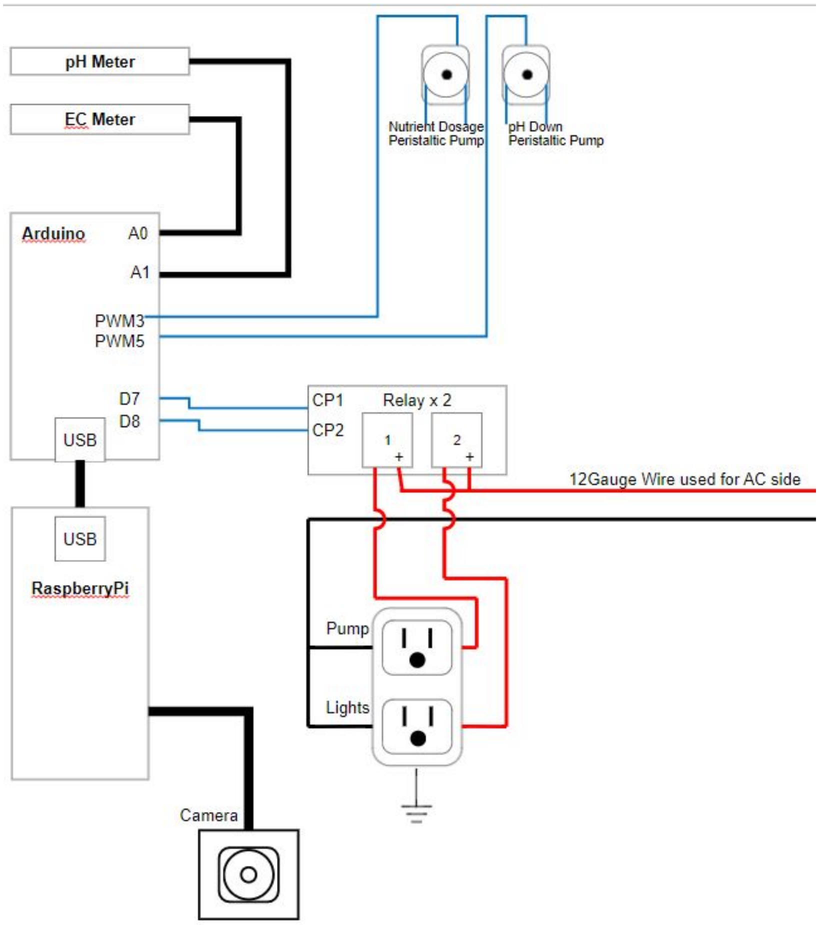


Figure 16: Low level circuit diagram for I/O components and microcontroller.

3.2.6 Persistence -- Database

Todo haris: do we remove this?

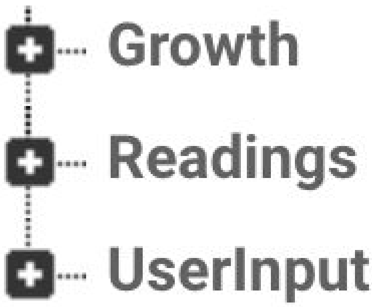


Figure 17: Firebase Database Root.

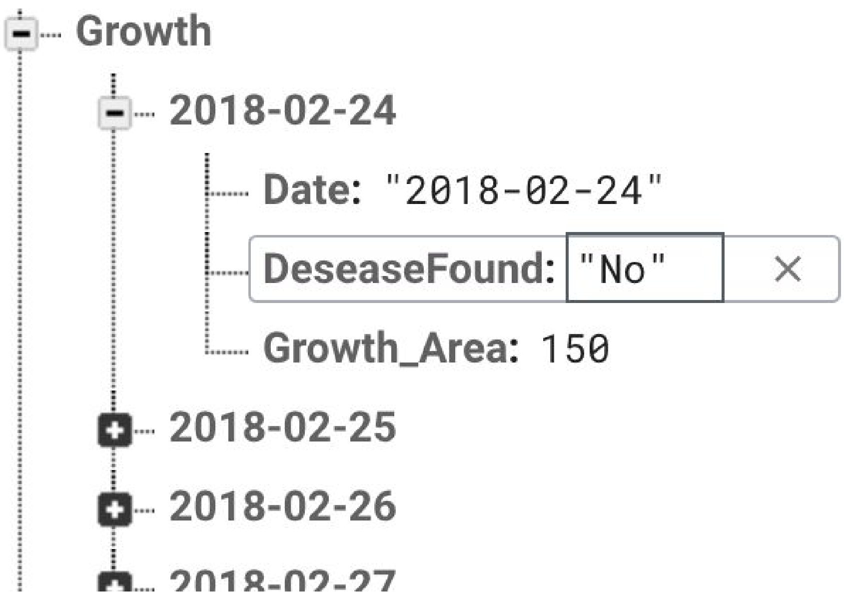


Figure 18: Growth

Growth Key is populated by the image processing program. This program runs once a day, takes a picture and runs image processing code. After the completion of image processing code, the values are pushed to the firebase database using a Python library Pyrebase(See Section 2.3.3 for more detail) .

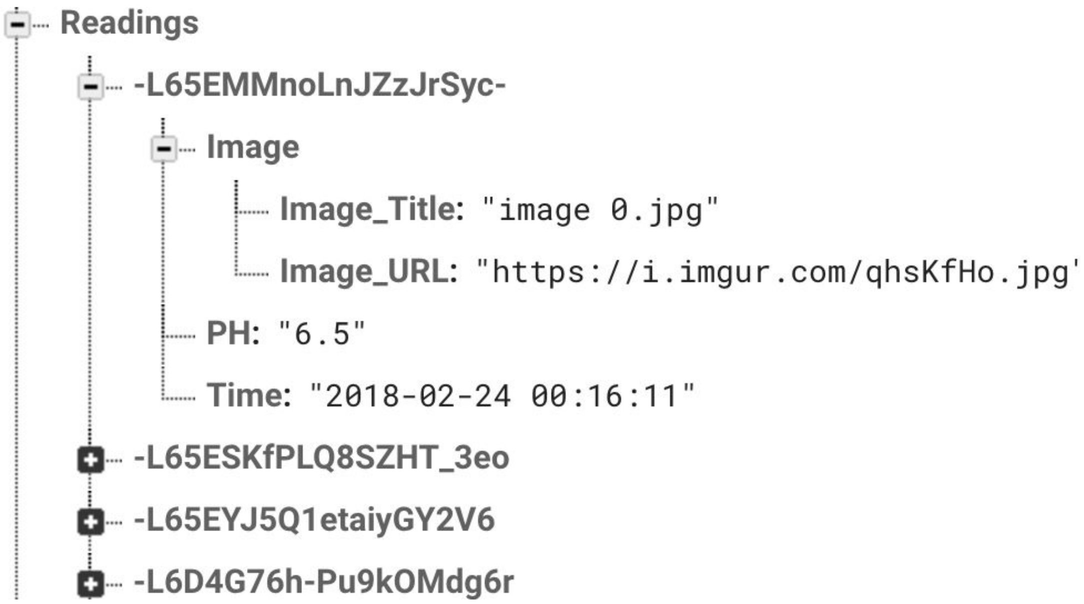


Figure 19: Sample readings from firebase.

Reading Key is populated by a program on Raspberry Pi that receives readings from Arduino on Serial Port. Once the readings are received every x time interval, the readings are pushed to Firebase database using a Python library Pyrebase(See Section 2.3.3 for more detail) .

3.3 Design Choices/Tradeoffs

Reference the wiring “up-to-code” setup.

3.3.1 Use of Microcontroller Arduino and Raspberry Pi

During our planning phase for this project, we were interested in making the communication between input/output devices to be simple. We originally considered using the Raspberry Pi alone to handle both the web service side, database, and communication between the sensors and actuators. This would be the simplest setup conceivable as there would be no need for a microcontroller. There was one major drawback to only having the Raspberry Pi, there was a lack of any analog I/O pins. We knew we would need these ports for our sensors for EC and pH, so it was decided to incorporate a microcontroller into the system. The microcontroller we decided to use was the Arduino Uno. This microcontroller supports analog and digital I/O ports, and can connect to the Raspberry Pi over a serial port connection (USB). There are 6 analog ports and 14 digital ports on the Arduino which will be plenty for our needs. Since the Arduino does have limited ram (2kB) and limited storage space (32kB) we decided to use the Arduino to handle interpreting and transmitting the I/O data to the Raspberry Pi.

To handle the communication over the serial port between the Raspberry Pi and the Arduino, we tried to make the communication robust by sending acknowledgment packets to the sender. This was done as a precaution in case a message failed to be received. The sender will only know if the message was received, and will not try and resend the message. The failure to receive an acknowledgement message will be logged for later use.

3.3.2 Use of Firebase and Imgur

Firebase is a back end platform for mobile and web application. It provides numerous services. However, depending on the kind of system, only the subset of services provided by Firebase are used in most cases.

We also used Firebase for its following services:

2.3.2.1 Firebase Realtime Database

Reasons:

1) Firebase provides a NoSql back-end database which is ideal for our project. The reason NoSql is ideal is because we can constantly add new sensor reading without having to redesign our schema. For example, right now our readings are coming from Ph Meter and EC meter. In the future we can easily add readings from a light sensor module.
2) It is accessible from anywhere. It can be simultaneously accessed from the Raspberry Pi as well as the Client sided application.
3) It is Real time. Firebase provides various APIs to sync your data at both ends. The APIs are designed in such a way that a client needs not to constantly check for any updates. Firebase developer have exploited Javascript event driven architecture.

4) Database Rules: Firebase Allows rules to be defined for specific user(s) for read/write purposes.

3.3.2.2 Firebase Authentication

We did not have to code any server sided user authentication code and user registration. Firebase provides APIs to register a new user as well as allow the user

3.3.2.3 Firebase Hosting

Firebase also provides a hosting space to host your application.

3.3.2.4 Imgur vs Firebase

Firebase storage space comes at a cost while Imgur is a website built primarily for sharing images. They allow unlimited amount of images uploaded to their website, of course at some limit per day. For the purposes of our project, we were not exceeding that limit and therefore decided to use Imgur for uploading our images. The uploaded images URL’s are stored in firebase database as shown in Figure.

3.3.3 Use of Pyrebase

Pyrebase is a Python wrapper library for accessing services provided by Firebase. This is the main library used in the project on the Raspberry Pi to communicate with the Firebase for the following purposes:

1) Push Data to Firebase Database
2) Listen for changes on Firebase Database values that are changed by the user inputs from the web or mobile app

3.3.4 Use of OpenCV Python Library

OpenCV is considered to be the most exhaustive libraries for image processing. It has the most amount of documentation and tutorials. Although it is not simple unlike many others but due to its mature age in image processing field, this library was chosen.

3.3.5 Use Of Front End Web/Mobile frameworks

Initially, a web application was to be developed which would be able to be opened from web and mobile browsers as well. To make the web app responsive on mobile web browser, Bootstrap Grid System was tol be used.

However, a decision was made to choose a mobile hybrid framework in order for the application to be deployed on multiple platforms.

The two frameworks that were considered were Xamarin.Forms and Ionic.

Xamarin.Forms is an API that allows building native applications for iOS, Android and Windows phones using a single codebase. This greatly reduces developer's time. However Ionic allows building applications not only for mobile devices, but for web browsers too. This also requires a single code base. However, one drawback of Ionic is that it is built on hybrid html, while Xamarin.Forms allows complete native code. Hybrid html applications are slower compared to native applications. The differences are summarized in the table below:

Feature(s)	**Ionic**	**Xamarin.Forms**
iOS Support	Yes	Yes
Android Support	Yes	Yes
Browser Support	Yes	No
Direct Native Code -> Faster	No → Slower Performance	Yes → Faster Performance
Documentation and Tutorials availability	A lot of documentation	Less Popular

Table 4: Xamarin vs Ionic

Even though Xamarin.Forms are better in terms of performance, Ionic is good for small scaled application. Therefore, it was decided to chose Ionic over Xamarin.Forms.

3.3.6 Use of CanvasJs

CanvasJs is a data visualization library for Responsive Web Apps. This is used primarily to show plant growth data as shown in Section 2.4.3.

3.3.7 Cost

Special Components Required:

Required Components - Automated Hydroponics System

Bottom unit could be a food grade plastic container, to act as a nutrient reservoir. (**$25**)

Microcontroller $**39.99** (Raspberry Pi Zero Budget Pack):

https://www.buyapi.ca/product/raspberry-pi-zero-budget-pack/

50W LED Grow Panel **$45**
https://www.amazon.com/KingTM-Spectrum-360-870nm-Greenhouse-Growing/product-reviews/B00V80UQR8/ref=cm_cr_arp_d_hist_5?ie=UTF8&filterByStar=five_star&reviewerType=all_reviews&pageNumber=1#reviews-filter-bar

pH Solution Powder for Quick and Easy Calibration **$12.99**
https://www.amazon.ca/XCSOURCE-Solution-Accurate-Calibration-BI672/dp/B06XDM843L/ref=sr_1_1?ie=UTF8&qid=1523416166&sr=8-1&keywords=ph+solution+powder+for+quick+and+easy+calibration

3-Way Elbow PVC Fitting, Furniture Grade, 3/4" **$30.06**
https://www.amazon.ca/FORMUFIT-F0343WE-WH-8-3-Way-Fitting-Furniture/dp/B00MNIYYQS/ref=sr_1_1?s=hi&ie=UTF8&qid=1523416225&sr=8-1&keywords=3%2F4+inch+size+3-way+elbow

Mylar Film, Plants Garden Greenhouse Covering **$12.99**
https://www.amazon.ca/NAVAdeal-Reflective-Greenhouse-Covering-Sheets/dp/B00QMOKCO2/ref=sr_1_1?s=hi&ie=UTF8&qid=1523416336&sr=8-1&keywords=mylar+film+plants+210

4-Channel Relay $**11.24**
https://www.buyapi.ca/product/4-channel-relay-module-for-arduino-raspberry-pi-5v/

Airline Tubing **$7.00**
https://www.amazon.ca/Python-25PAL-25-Feet-Airline-Tubing/dp/B000255NYQ/ref=sr_1_2?s=hi&ie=UTF8&qid=1523416545&sr=8-2&keywords=25+foot+airline+tubing

3x 12V DC Peristaltic Dosing Pump **$51.00**
https://www.amazon.ca/Yosoo-Dosing-Peristaltic-Aquarium-Analytical/dp/B01D0WQYJU/ref=sr_1_6?s=hi&ie=UTF8&qid=1523416514&sr=1-6&keywords=12v+dosing+peristaltic

Temperature & humidity sensor **$15.00**
https://www.buyapi.ca/product/am2302-wired-dht22-temperature-humidity-sensor/

Submersible Temperature Sensor **$25.00**
https://www.adafruit.com/product/642

pH sensor: **$40.00**
http://www.robotshop.com/ca/en/gravity-analog-ph-meter-kit.html

3x MOSFET, 3x 1k Ohm Resistor, 3x Diode **$20** http://www.Digikey.ca

Misc. cables

Raspberry Pi Camera
https://www.buyapi.ca/product/raspberry-pi-8mp-camera-board-v2/?src=raspberrypi
34.95 CAD

Rasspberry Pi Camera Cable
https://www.amazon.ca/gp/product/B00I6LJ19G/ref=ox_sc_act_title_1?smid=A32EQF38HXAF7A&psc=1
9.99 Cad

Raspberry Pi 3 B Starter Kit
https://www.buyapi.ca/product/raspberry-pi-3-b-starter-kit/
78.95 CAD

Total: $ 458.94

3.4 Results and Accomplishments

The following sections provide overview of the results obtained in this project

3.4.1 Plant Growth

Plant Growth Algorithm with the help of OpenCV functions

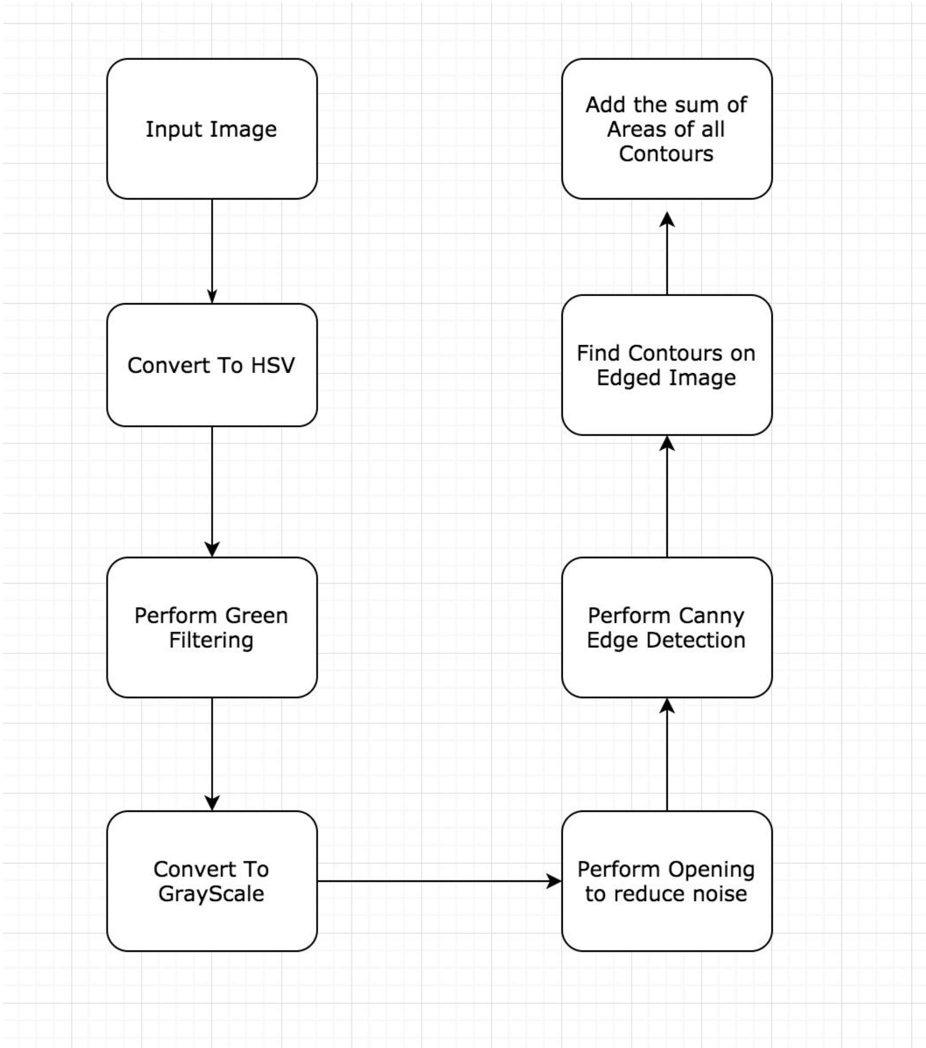


Figure 20: Plant Growth Algorithm

Step 1)

Figure 21: Raw Image

Step 3)

Figure 22:Green Filtered

Step 5)

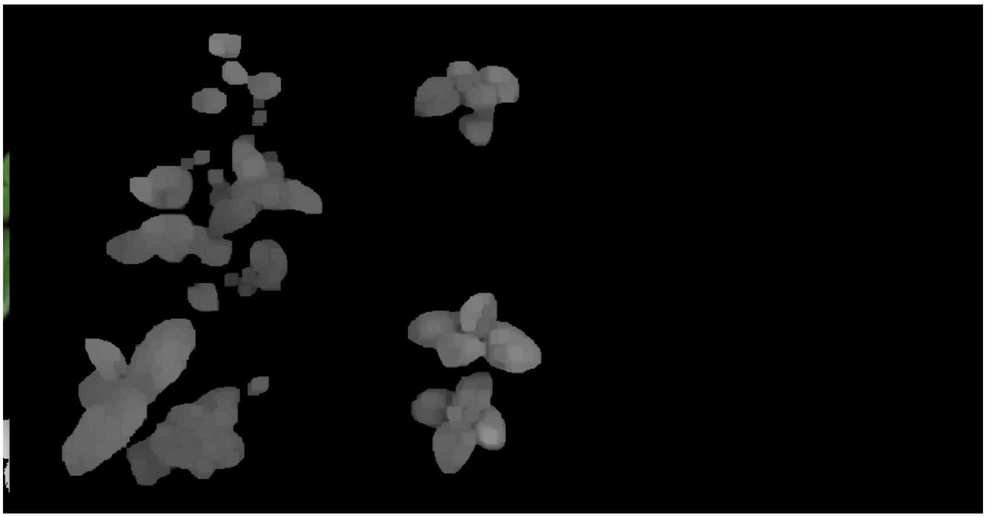


Figure 23: Grayed

Steps 6,7,8)

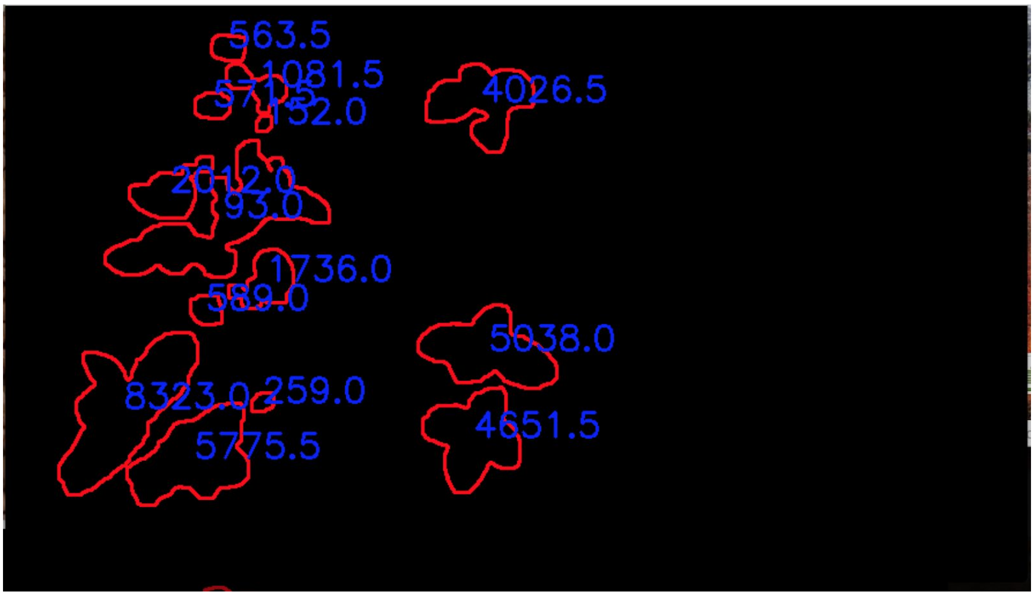


Figure 24: Contours with their respective Areas

3.4.2 Disease Detection

Disease Detection can be done by detecting contours within contours and the hierarchical relationship between those contours. The same algorithm is applied to detect disease except when looking for contours in image, its hierarchical relationship is also obtained. Then, out of all the contours, it is determined whether they are convex or not. The test subjects that are included in the experiment were as follows.

When looking for contours, only this line makes a difference to get contours hierarchical results:

#_, contours, _ = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)

_, contours, _ = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)

The blue line(the first line) looks for only external contours while the green line(second line) looks for contours as well as their relationship with other contours. This makes it possible to obtain parent - child relationship between contours.

Plant without disease:

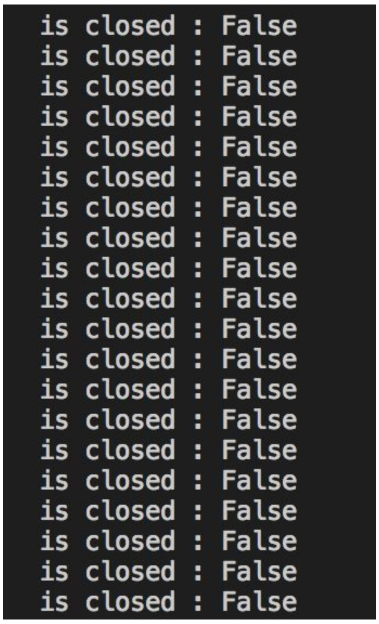
Image	Contours	Result
		 is closed : False

Table 5: Plant without Possible Disease

Plant with Possible Disease:

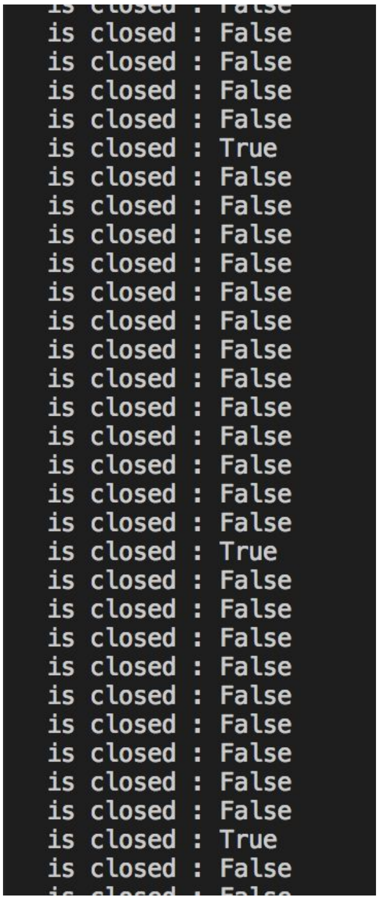
Image	Contours	Result
		

Table 6: Plant with Possible Disease

3.4.3 Web UI

Section 3.4.3.1 Login

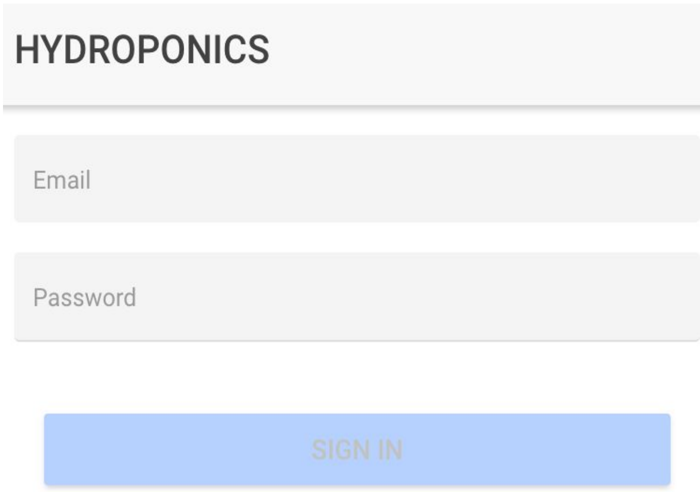


Figure 25: Web or Mobile Login

Section 3.4.3.2 Controls

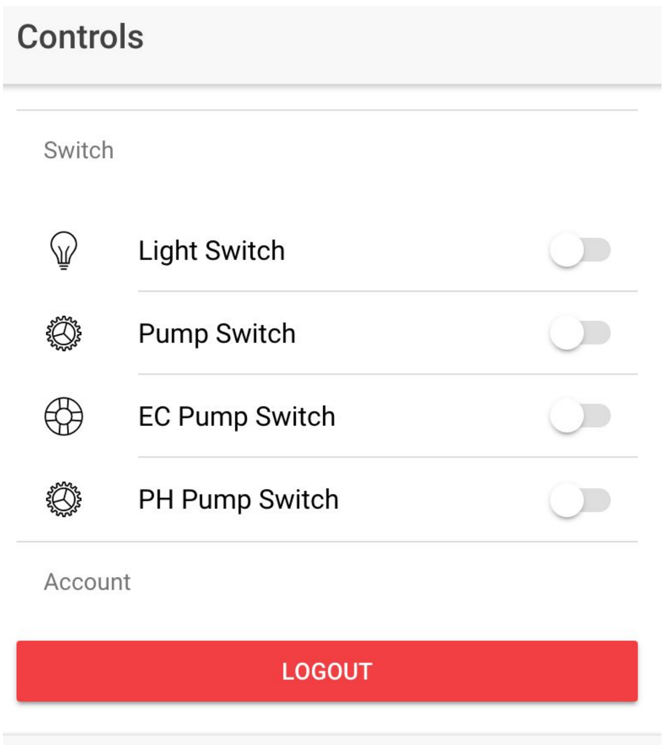


Figure 26: Web or Mobile Controls

Section 3.4.3.3 Live values:

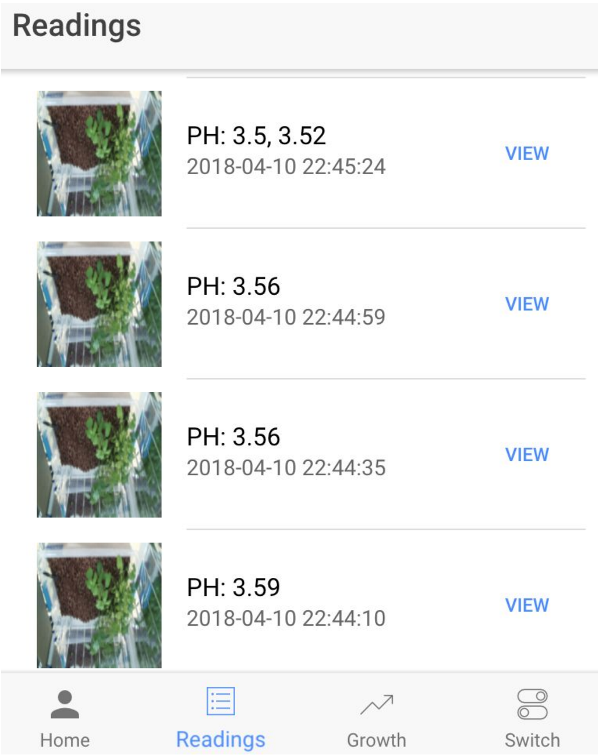


Figure 27: Live Values

Section 3.4.3.4 Growth Graph

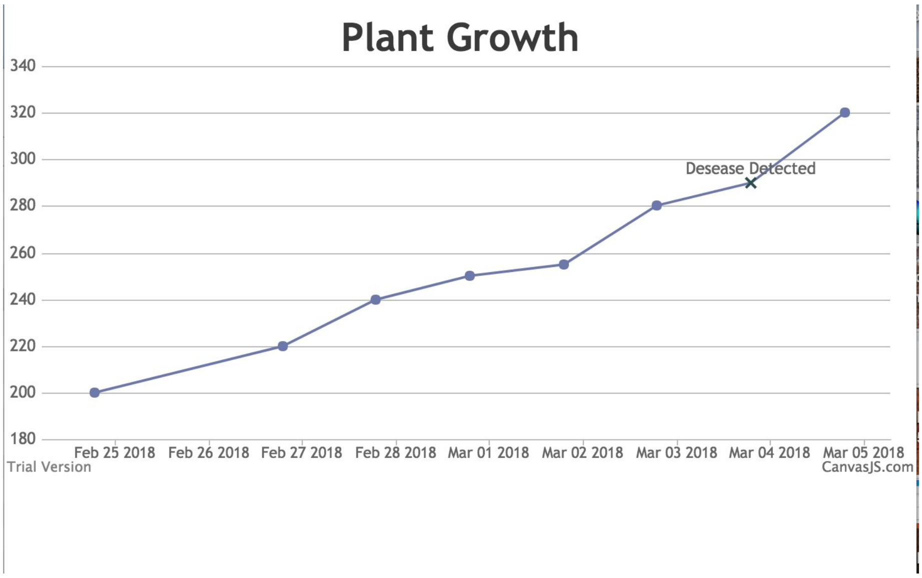


Figure 28: Plant Growth line graph

3.4.4. Serial Communication and Robustness

The serial communication between the RaspberryPi and the Arduino was done in order to send necessary commands and data between the two devices. To add robustness to this communication, acknowledgement messages are sent from the Arduino to the Pi once a command is received. The Pi will wait 3 seconds before trying to retransmit. This is only a one way acknowledgement as the data that is being sent from the Arduino to the Pi is not very critical, and consists of sensor data. This data is sent every 20 seconds regardless and therefore does not require validation that it was received every time. By using the acknowledgements, we have achieved a more robust communication for the Pi to send commands to the Arduino.

3.4.5. Time Synchronization of Arduino

The Arduino isn’t connected to the internet, and therefore cannot know the current time when first turned on. This is an issue when when setting up the schedules for the lights and the pumps. This is overcome by using the Pi to send the current time data through the serial port to the arduino, which then updates its clock. This allows the Arduino to be turned off and on without affecting the growing cycles, and adds robustness to the project.

4. Conclusion, Recommendation and Maintenance

4.1 Conclusion and Recommendations

The goal of this project was to create an automated hydroponic system that is easy to use for novice growers with little to no experience. This will supply edible produce in a safe and convenient way, and also provide user data about the growth rate of the plants and notifying the user if there is a potential diseased area on the plant. This was accomplished by using an Arduino to control the actuators and sensors, and a RaspberryPi to orchestrate commands and data between the microcontroller and the database/web app. The framework and technologies used were carefully selected in order to produce a useful application which can be deployed on multiple user platforms (iOS, Android OS, and web).

The current state of the system provides enough features to allow a user to grow a variety of plants with very minimal input into the system. The user can control all the actuators via the web application, and monitor the growth of the plants and view pictures of the plants online. Some suggestions for further development on the project would include: physical buttons on the system which allow the control of actuators of the system, and more features to display and track data on the web application. Robustness could also be added to the RaspberryPi in the event of a power failure, such that the system reboots and the necessary scripts are correctly executed, preserving the integrity of the plants.

Future work on the project could focus on machine learning to improve subsequent growth cycles, increase the dynamic behaviour of the system, as well as better disease detection. This would increase the efficiency of the system and increase the yields for the user.

4.2 Maintenance

The repository is available at the following url:
https://github.com/hersh94/IoT-Hydroponics

5. References

<!-- Reference will be modified as these are not up to date --!>

References:

[1]: Barbosa, G. L., Gadelha, F. D., Kublik, N., Proctor, A., Reichelm, L., Weissinger, E., . . . Halden, R. U. (2015, June). Comparison of Land, Water, and Energy Requirements of Lettuce Grown Using Hydroponic *vs.* Conventional Agricultural Methods. Retrieved September 27, 2017, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4483736/

[2]: Skura, E. (2016, June 24). Food in Nunavut still costs up to 3 times national average. Retrieved September 27, 2017, from http://www.cbc.ca/news/canada/north/nunavut-food-price-survey-2016-1.3650637

[3]: Khirade, S. D. (n.d.). Plant Disease Detection Using Image Processing. Retrieved September 27, 2017, from http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7155951

[4]:Garfield, L. (2017, October 02). Ikea has debuted an indoor farm that grows greens 3 times as fast as in a garden. Retrieved October 04, 2017, from http://www.businessinsider.com/ikeas-space10-designed-a-vertical-farm-for-the-home-2017-9

[5]: A., & I. (2016, May 12). Hyduino - Automated Hydroponics With an Arduino. Retrieved October 04, 2017, from http://www.instructables.com/id/Hyduino-Automated-Hydroponics-with-an-Arduino/

[6]: (n.d.). Retrieved April 9, 2018, from http://www.motorleaf.com/

[7]: Ontario Electrical Safety Code. (n.d.). Retrieved April 9, 2018, from https://www.esasafe.com/contractors/the-ontario-electrical-safety-code

[8] Ontario Electrical Safety Code. (n.d.). Retrieved April 9, 2018, from https://www.esasafe.com/contractors/the-ontario-electrical-safety-code

[9]Ontario Electrical Safety Code. (n.d.). Retrieved April 9, 2018, from https://www.esasafe.com/contractors/the-ontario-electrical-safety-code

For general knowledge in terms of creating a self sufficient food source.
And, D. H. (2001, November 01). David Heber. Retrieved September 27, 2017, from http://jn.nutrition.org/content/131/11/3078S.short "...one person working **1 acre can feed 120 people".**

[10]: (n.d.). Retrieved April 10, 2018, from http://www.kevblog.co.uk/create-3d-array-in-actionscript-3/

[11]: (n.d.). Retrieved April 10, 2018, from http://www.kevblog.co.uk/create-3d-array-in-actionscript-3/

[12]: (n.d.). Retrieved April 10, 2018, from https://www.processing.org/tutorials/color/

[13]: (n.d.). Retrieved April 10, 2018, from https://en.wikipedia.org/wiki/HSL_and_HSV#/media/File:HSV_color_solid_cylinder_alpha_lowgamma.png

Useful Links:
Git Repo: https://github.com/hersh94/IoT-Hydroponics

http://ceur-ws.org/Vol-1498/HAICTA_2015_paper28.pdf

https://www.arduino.cc/en/Reference/Board

Tools used for diagrams:
https://www.draw.io/