|  |
| --- |
| Otago Polytechnic |
| PlantMate |
| Smart Irrigation System + Environmental Monitor |

|  |
| --- |
| Samuel Batchelor  [Date] |

Contents

[Introduction and Explanation of Project. 1](#_Toc200834051)

[User Stories. 1](#_Toc200834052)

[Basic Research. 3](#_Toc200834053)

[Hardware Used. 7](#_Toc200834054)

[Software Libraries. 15](#_Toc200834055)

[Insights and Discoveries. 16](#_Toc200834056)

[Custom software 24](#_Toc200834057)

[Future Scope. 25](#_Toc200834058)

[Final Project. 25](#_Toc200834059)

[Circuit Diagram. 26](#_Toc200834060)

# Introduction and Explanation of Project.

*Write about half a page.*

*Explain how you came to design and build your project. What is the original inspiration for this project? Why does this topic matter to you. This section can be finished last if necessary*

PlantMate is a smart irrigation system and environmental monitor project that automatically waters a plant and provides handy environmental data about a plant, to keep the user informed about a plant’s needs.

The whole idea for this project, is mainly because I am simply into plants. I have been owning and nurturing a growing collection of plants for the last 4 years now, and one of annoying things I came across was watering them. It became a real pain when I had to go down the back of my house to drag the garden hose up, each time I wanted to water my plants at the front of my house. When I found out IT and plants could be linked together, I was more than ecstatic. In my second embedded project when I was searching for plant-related project ideas, I came across the concept of a smart irrigation system. And thus, PlantMate was born. In my third project I wanted to extend the scope and size of PlantMate even further and include the water pump, as well as add a few more sensors and get PlantMate onto the cloud too.

# User Stories.

*Write about half a page.*

*Write as many user stories as you can to support your project. It would be ok to have user stories that don’t get implemented (But maybe explain why).*

*Once you’ve done that, define some use cases or functionality that will support the user stories you have chosen to implement.*

*The correct format is:*

*As a [type of user] I want to [describe need] so that I can [reason]*

*As a gamer I want a way to easily play any game I own because I can never find the original media*

*As a gardener I want to apply just the right amount of water so I don’t kill my plants*

*Remember user stories don’t provide technical solutions!*

*This answers the basic question of “who” is a user of your project.*

**As a plant owner I want to ensure that my plants are being well watered, know when to water them and ensure their needs are being met.**

**As a plant owner I want to be able to water my plant remotely through a website, so I don’t have the hassle of watering them whenever I am at home.**

**As someone who works in a garden centre, I want to gather valuable environmental data about plants so I can get a general understanding of their environmental needs so I can explain them to someone who is considering buying a plant.**

**As a someone who works in agriculture, I want to give my crops a precise amount of water so I can maximize my crop yields.**

**As someone who is propagating seedlings, I want to implement a watering schedule for all the seedlings so I can ensure they are getting the care they need.**

**As a forestry worker, I want to gather data from multiple plants of a certain species, to better understand the environmental requirements and needs of that species.**

# Basic Research.

*Write about a page.*

*You should preface all work with some form of simple research. The goal of this is to be able to make informed decisions and justify choices that you have made for your project.*

*This can be checking for sensors or devices from a range of suppliers based on features, simple comparison of software, etc. You can also discuss some of the wider issues that might exist in the problem domain, what is in scope and out of scope (and why you think so).*

*The point of this is to answer the question of “why”. Why did you choose these components, why you did your project this way and not another, why your project is a good solution to the problem, etc…*

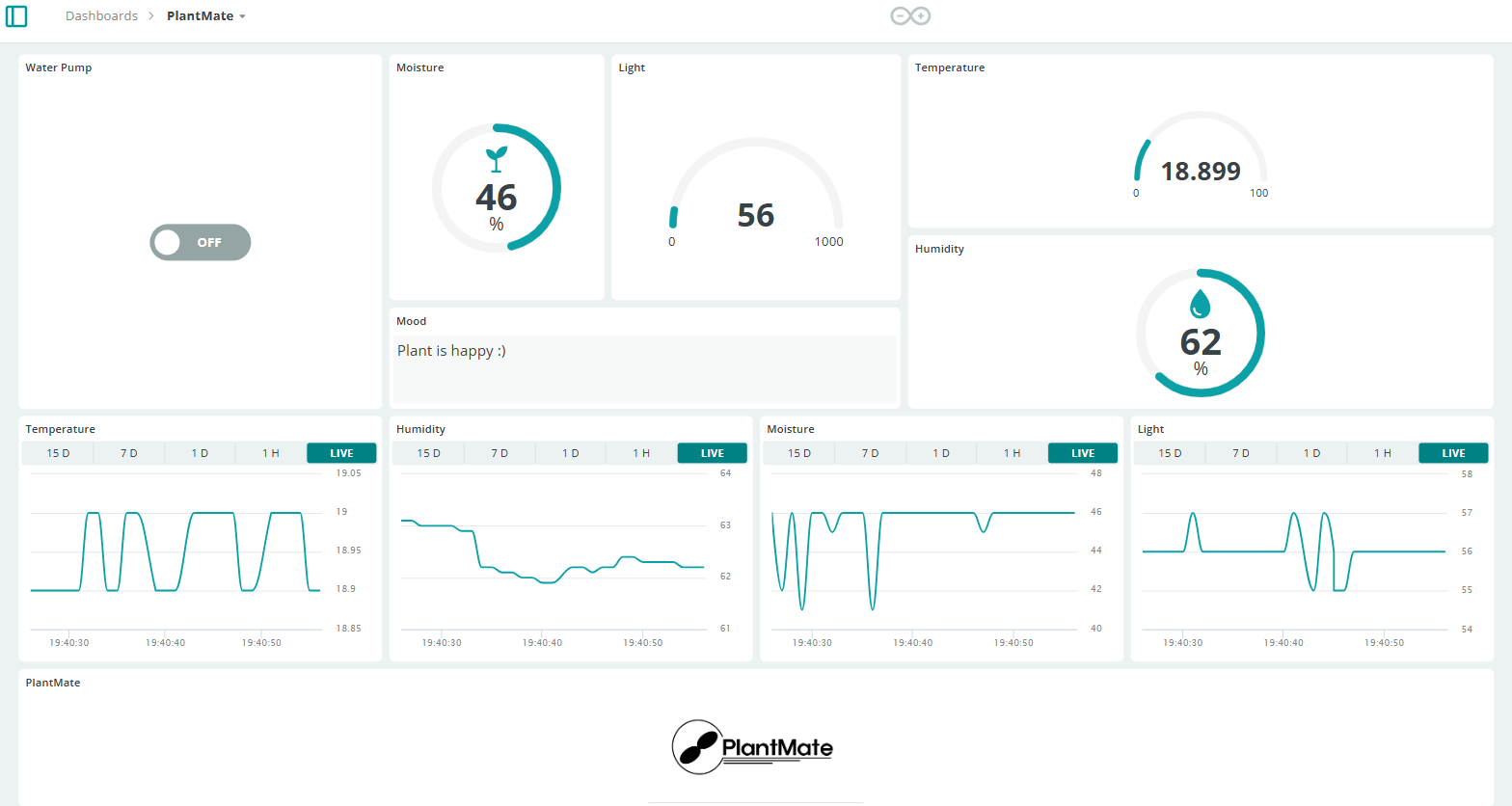
**Initial Research**

When finding out how to make a smart irrigation system, I realized that most people used soil moisture sensors, temp/humidity and water pumps for their projects. But the thing I wanted was to expand the scope by a slight amount, because since I’m a plant person I took into consideration other environmental factors a plant is subject to. One factor I thought about was light, and of course I did some research on what do for that.

**IOT:**

I was intrigued by one person’s smart irrigation project where they used the Arduino IOT Cloud to turn on their water pump to water their plants, right from the cloud.

I couldn’t help myself but find out how to implement it for my project.



**Temperature and Humidity**

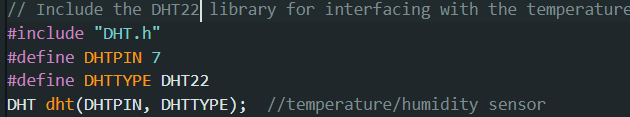
When I was digging around doing some research for an ideal temperature/humidity sensor for this project, I stumbled upon the DHT11 sensor, as it was an easy lightweight sensor I could use to gather relative temperature and humidity. I found out the **Jaycar** Dunedin store had some DHT11’s in stock. So I went and ordered a DHT11 sensor from **Jaycar,** which was for my previous embedded project.



Later, when I had all my 4 of my sensors hooked up to my Arduino, somehow my DHT11 overheated, and I quickly had to think of an alternative. After finding not only a better replacement sensor, I did some research to compare it to the DHT11 sensor. The research showed that the DHT22 was a more costly but more accurate bigger brother than the DHT11 sensor.

**Libraries:**

I am currently the using Adafruit’s **‘DHT’** library for interfacing with the DHT22.

****

**A screen shot of a computer program

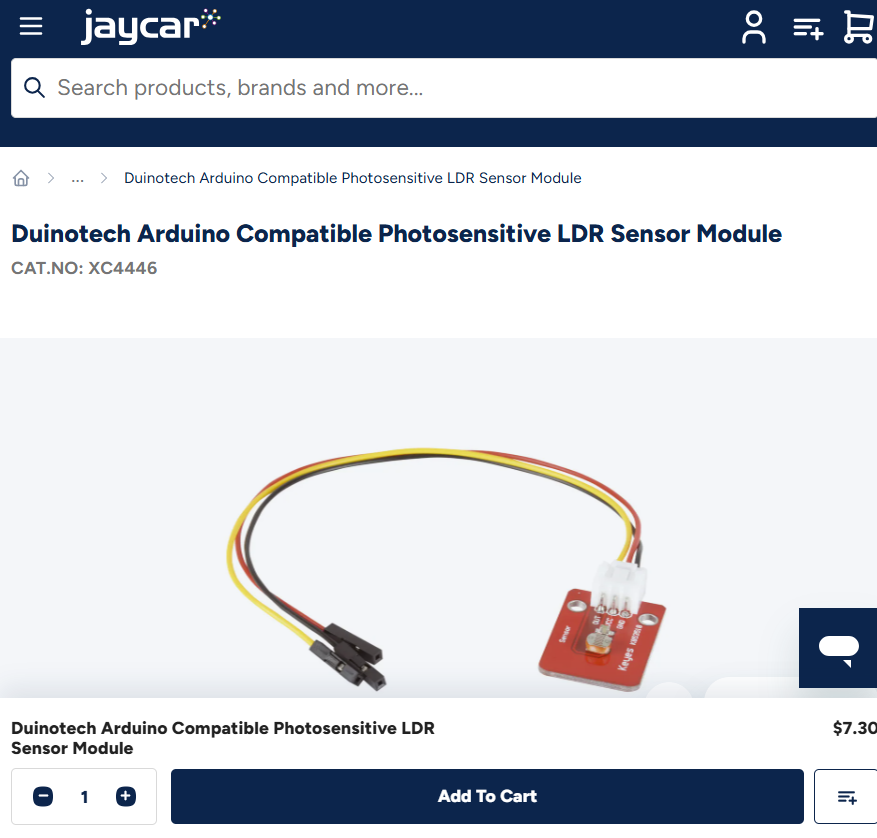
AI-generated content may be incorrect.**

**A computer code with orange and white text

AI-generated content may be incorrect.**

**Light**

When doing research for an ideal light sensor, I came across something called a Photosensitive LDR Sensor. And once again I got this sensor from Jaycar. I gathered that the sensor detects resistance based on light intensity. It seemed that the lower the light intensity, the higher the resistance.



**Soil Moisture**

**Resistive Soil Moisture Sensor Module**

When doing research for smart irrigation project ideas, I noticed that a lot of people used a resistive soil moisture sensor, since its cheap and gets decent readings.

After I ordered a resistive soil moisture sensor from Jaycar for my previous embedded project, I grabbed code examples on how to convert the raw analogue readings from the sensor into more human friendly percentage values.

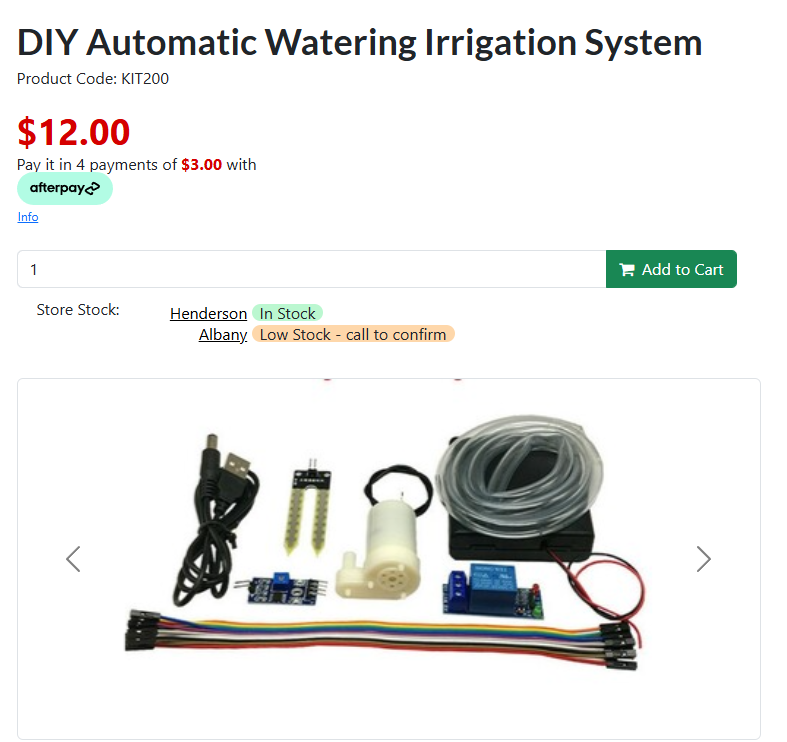


When doing some further research on resistive soil moisture sensors, I realized they weren’t as reliable as I thought they were. It turns out resistive soil moisture sensors are prone to corrosion and have a shorter lifespan. So, I looked for a better more accurate, but more expensive soil moisture sensor. I ordered a capacitive soil moisture sensor to not only have a more accurate sensor but to have a sensor that is less prone to corrosion.

**Note: I ordered the capacitive sensor from an online store called Fruggo**

**Water**

When looking at some smart irrigation projects, people used water pumps that had output voltages of between 5 to 12 volts. And I also noted that these pumps were controlled using a relay module. So, I ordered a smart irrigation kit from a store called Surplustronics, which not only consisted of a pump and a relay, but also an external power supply for the pump too, and a few other goodies.

****

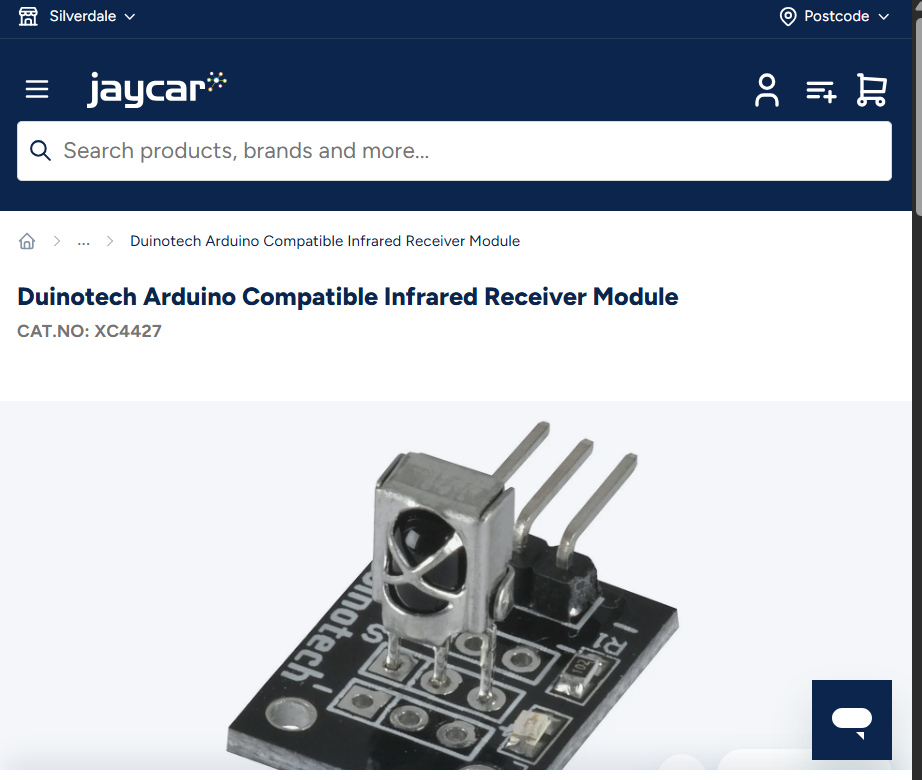
**How to implement a menu system**

**Initial menu system (Second project)**

When I did my second embedded project, I had a button system that consisted of two push buttons to change the display on the i2C LCD I was using. This way I had somewhat of a basic menu system. I got this button menu idea from a smart irrigation system project someone at Polytech made last year in 2024. The first button was for going to the temp/humidity menu and the second button was for going to the soil moisture menu. Now once I started developing my current project, for a while I had contemplated implementing a similar button system, but overtime as I added more and more sensors to my project, there was less pin space on my Arduino. A few of the cons of this button system, is that I would have to have sufficient room on my Arduino to compensate for the button pins. The wiring would also be quite a mess too, and there would be a lot more objects drawing power from the Arduino too.

**New menu system**

So, a couple weeks ago, I found a very good alternative. I decided to implement a remote-control system which would consist of an infrared remote and receiver. But to implement this, of course I had to find out how this would work. From what I gathered, it seemed when any button on the remote was pressed it emitted an infrared beam. Then the receiver would pick that beam up and with the aid of a software library, decode the beam. The beam would be decoded to a unique hexadecimal address. So essentially the remote system as about mapping unique hexadecimal addresses to buttons, to identify each button. I not only had to choose which buttons on the remote I was going to use but, I had to map their hexadecimal addresses picked up by the receiver.

****

I did some digging on Jaycar’s website and purchased an infrared receiver. But as discussed later in this narrative, things didn’t go quite as expected with this sensor.

# Hardware Used.

***Write more than one page****. Include relevant pictures with explanation.*

*List and justify your hardware choices here. Explain how these fulfil the requirements of your project, how they will implement use-cases based on your user stories.*

*Explain to me how your project is put together. Use lots of clear relevant pictures, making sure they have an explanation about what they are showing.*

*Use this section to provide one or more clear diagrams of the layout of your components in your project.*

*List the technical requirements your project has implemented (See assessment instructions)*

***I should be able to use this section to follow along and build your project from scratch.***

*This answers the basic questions of “what” and “how”*

**Arduino Uno WIFI R4**

To implement the IOT part of my project and push this project to the cloud I decided to purchase my self an Arduino Uno WIFI R4 board from Jaycar.

**A computer chip on a website

AI-generated content may be incorrect.**

****

This meant I had to officially retire the Arduino Uno I was using for most of my time in the embedded systems paper, because of course it didn’t have a built-in Wi-Fi capability.

**DHT22 Temp/Humidity Sensor**

A close-up of a device

AI-generated content may be incorrect.

After my DHT11 sensor I used overheated, I investigated on not only getting a new replacement, but something that was more accurate.

After doing some research I decided to go for the DHT11’s bigger brother the DHT22, which has a wider temperature and humidity range. This was true when I realized the DHT11 readings mainly had a greater margin of error compared to those of the DHT22. When using the DHT22 I got a much better temperature range which seemed to be a lot more accurate for the environment the sensor was in.

A screenshot of a computer

AI-generated content may be incorrect.

**How are they different?**

* Temperature Range
  + DHT11: -20 to 60℃
  + **DHT22: -40 to 80℃**
* Temperature Accuracy
  + DHT11: ±2%
  + **DHT22: ±0.5%**
* Humidity Range
  + DHT11: 5 to 95% RH
  + **DHT22: 0 to 100%RH**
* Humidity Accuracy
  + DHT11: ±5%
  + **DHT22: ±2%**

The sensor will not only just give basic temperature and humidity readings, but it will also aid in giving the user a clear picture of what a plant’s environment is like. In a scenario where the climate for a potted plant is not ideal, the user can compare the DHT22’s values to the recommended temp and humidity values of a plant and perhaps move the plant to a more ideal location.

**Capacitive Soil Moisture Sensor v1.2**

A hand holding a small device

AI-generated content may be incorrect.

After the readings of my original resistive soil moisture sensor, were getting less accurate with time, I swapped it out with my new capacitive sensor. This sensor will not only be more rust resistant, but it will provide better more accurate moisture readings for the user to read.

**Original Resistive Soil Moisture sensor from Jaycar:**



When the capacitive sensor senses a lack of moisture, the value peaks around in the late 500s, and conversely when there is moisture the value is in the 200s

Switching my old resistive sensor for the new capacitive sensor meant I had to update my code. I had to change the low and high sensor values and invert the mapping of the values to a percentage.

**Note: it used to be 0 to 100, now its 100 to 0**

A screen shot of a computer code

Description automatically generated

The soil moisture will be able to fill the user requirements by providing the user quality data about how dry or wet their plant’s soil will be. The sensors raw analog readings are mapped within a percentage range of 0 to 100%, with 30% being low and 70% and above being wet. From the user will get a rough sketch of the moisture levels their plant is enduring.

**5V DC Water Pump**

The water pump I am using runs on an external 12V power supply with 4 AA batteries.

The ground of the power supply is connected to that of the pump. The VCC of the power supply is connected to the COM port of the relay I’m using in this project. The VCC of the pump is connected to the NC port of the relay. Through the relay I can turn the pump on or off, because of course with relays, they allow you to communicate with high voltage devices. This pump will be a major asset for the user where they can either manually toggle the pump on or off or let the automation of the project do it instead. Assuming the user wants to toggle the pump manually, this would usually be if the user wants to have better control over a plant’s watering schedule instead of completely relying on automation.

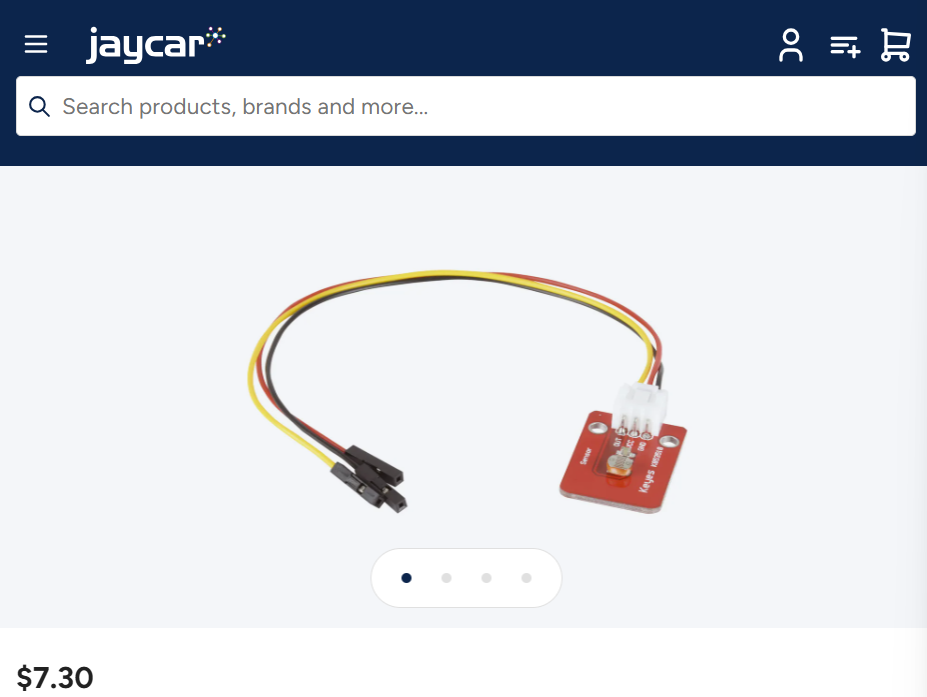
A white box with a black cord attached to a white box

AI-generated content may be incorrect.

**Photosensitive LDR Sensor Module**

The sensor I am using to measure light levels is a photosensitive LDR sensor, which uses a light dependant resistor that produces an output voltage that fluctuates with light level.

To ensure the readings are somewhat accurate, I must tilt the sensor relative to the box the project is in at a 45-degree angle. This is so I can aim the sensor at the sky, and the sun is able to travel along the sensor’s line of sight.

****A close-up of a metal tube

AI-generated content may be incorrect.

This sensor will give the user a rough idea of the light levels their plant is subject to. The web interface for this project will provide a visual of the varying light levels of a plant overtime. A plant owner can make sufficient use of this data by comparing the amount of light their plant gets vs the amount of light a plant of the same species should be getting. This will of course be another one of the methods of ensuring a person’s plant’s needs are being met. Light being crucially important.

**LCD Screen, with soldered i2C controller**

**A rectangular object with a blue screen

AI-generated content may be incorrect.**

This LCD screen will provide the user with an interface where they can view different plant needs, such as the temperature, humidity and moisture of course. The LCD has an I2C controller soldered onto the back of it, not only to have some form of I2C communication but have a lot less pins taken up on my Arduino UNO.

**Note: This is when the user is not using the cloud to view the data**

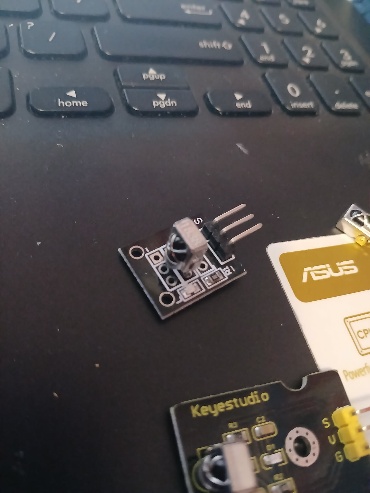
**SRD-05VDC-SL-C Relay module**

A blue electronic device with wires

AI-generated content may be incorrect.

The relay module will allow the Arduino Uno to control the water pump, as the water pump has a input voltage of just over 5V. When the relay is switched on the pump circuit will be live, triggering the pump. The NC port of the relay has the pump VCC going through it, and the COM port has the VCC of the pump’s power supply going through it.

**XC4426 TX Module and Car MP3 remote**

The Infrared receiver and remote come from the Arduino kit I was provided with. This remote system will allow the user to switch between different modes on the LCD screen. This will be guaranteed to be an easier system for the user to control PlantMate, instead of using a built-in button system which would have required soldering a few buttons to a copper board, then wiring them to the Arduino, which wouldn’t have been very efficient.

Unfortunately, when attempting to relocate my receiver, and accidentally wiring up the sensor wrong, I overheated it. So I ended up replacing my old VS1838b with a TX Module (XC4426)

A hand holding two wires

AI-generated content may be incorrect.

**HC-SR04**

**A hand holding a small device

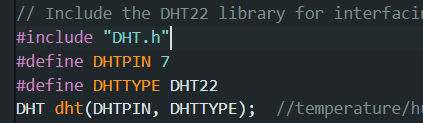
AI-generated content may be incorrect.**

The HC-SR04 will be utilized in this project to measure the water level in a water tub/container.

This way the user will know when the tub/container is empty, and they can for example either go home and refill the container or notify someone else to fill up the container.

# Software Libraries.

**Adafruit’s DHT library**



**Liquid Crystal I2C library**

**A close up of text

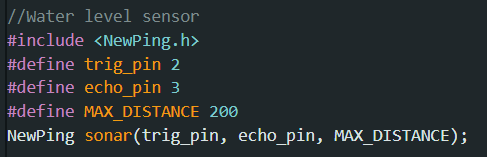
AI-generated content may be incorrect.**

**IRremote library**

**A close up of a text

AI-generated content may be incorrect.**

**NewPing Ultrasonic library**

****

**Custom analogue sensor library**

A screen shot of a computer program

AI-generated content may be incorrect.

# Insights and Discoveries.

*Finally, some reflection and self-analysis. Appraise your work, tell me about any insights you have gained from this, what you would do differently, what you particularly liked about your solution, etc.*

**Building a circuit of a few wires and copper tape**A box with wires and wires

AI-generated content may be incorrect.

When my previous wiring was driving me nuts, my new copper tape setup drove me even more insane when my soil moisture sensor readings where all up the pole. Also my DHT22 sensor readings were janky too, showing up as `nan`. So although the copper tape is conductive something obviously didn’t seem quite right when the soil moisture sensor readings were fluctuating weirdly. **Note: this was my resistive soil moisture sensor.**

**Weird issues with DHT22**

One of my main concerns is whenever I turned on the relay that powers my water pump, the readings from the DHT22 sensor would become corrupted.

Note: The temperature and humidity readings showed as ‘nan’ when the pump was being powered. This is assuming my pump was drawing to much current despite being connected to an external power supply.

A screen shot of a computer

AI-generated content may be incorrect.

I still occasionally get `nan` readings from the DHT22 whenever the pump is turned on, maybe it is due to pump drawing a bit current despite being attached to a relay.

**Resistive vs Capacitive Soil Moisture sensors**

**Less accurate readings**

Overtime my resistive soil moisture sensor readings were becoming less accurate and reliable. For example, when I dipped the two sensor prongs into water, the percentage value wasn’t 100% like it should’ve been. The value hovered around 70% to 80%

**Corrosion (maybe)**

As a result of using my sensor overtime and dipping it in soil occasionally it appeared that the two prongs started to have, possibly some slight oxidization on them. This is of course as I found out is one of the big disadvantages of having a resistive sensor like the one, I had.

**Depth of the Soil Moisture sensor**

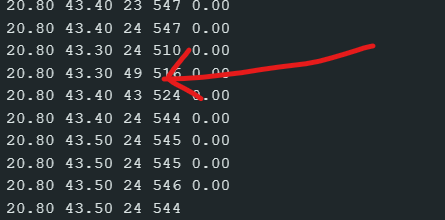
From what I discovered, it seemed like the soil moisture readings varied quite a bit depending on the depth of the sensor in soil. The deeper it seemed to be higher the readings, and conversely the less deep it was the lower the readings. This can obviously be quite misleading, so to work around that, it is good to place the sensor at the same depth each time it gets inserted into the soil. This will ensure more accurate and consistent results.

**A computer and a plant in a pot

AI-generated content may be incorrect.**

**Increase in soil moisture sensor readings**

All though it seemed like all good was going on with the sensor, I suddenly when placing my fingers around the sensor, the readings suddenly jumped. This would be due to the fact human fingers would have a higher dielectric constant than dry soil, and the sensor picking up changes in a dielectric constant.

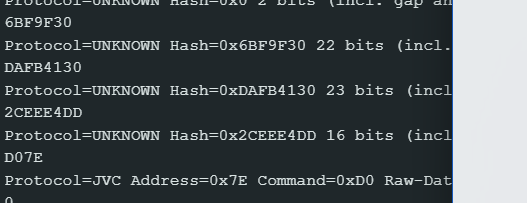


**Weird inconsistent hexadecimal readings produced from infrared remote messages**

One major problem I’ve had with the infrared receiver I ordered from Jaycar is that whenever I press a single button of a remote, there is a unique hexadecimal value generated every time, which is not normal. Every button should have its own unique hexadecimal value. Whereas when using the standard three-legged sensor, I get more consistent readings from a remote, with slight error of course.

***With the Jaycar bought sensor:***

*Note: I’m pushing the same button on the remote*



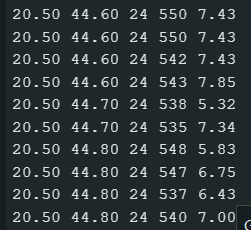
***With the provided sensor in my kit:***

*Note: for the first four readings I’m pressing button 1 on my Car mp3 remote and the last reading is from button 2 on my remote*

A screenshot of a computer program

AI-generated content may be incorrect.

**Mostly inaccurate readings from the HC-SR04**

****

The readings from the HC-SR04 aren’t the most accurate. When holding the sensor at a steady height, the readings can vary slightly. Sometimes, I can get the occasional spikes in readings.

Another big problem with measuring water level is that depending on the height of the water container, the maximum height will vary. E.g. going from a 20cm tall container to 40cm tall container, will mean in the code the value for holding maximum height will have to change.

**Can’t use remote when pump is running**

Once again possibly due to electrical interference, whilst the pump is running, I can’t emit infrared signals using the remote to toggle different menu states on PlantMate. So, I have contemplated scrapping the feature where the user can turn on pump with the remote, due to this interference issue.

**A screen shot of a computer

AI-generated content may be incorrect.**

**Having similar but slightly separate local and cloud-based codebases.**

Whilst I was in the process of deploying PlantMate to the cloud, I came across a big problem. I couldn’t just use my local code on the Arduino IDE, but I had to have cloud-based code too. The cloud-based code had some properties that were generated by the Arduino Clouds online builder environment. Also the items that I added on to the dashboard such as the gauge for light levels, each has their own global variable. So the light gauge has a global integer variable called light. The switch for toggling the pump on and off, is attached to a global boolean variable. Each time a new global variable is created, it is automatically added to the ‘.ino` files, and other associated files on the cloud end.

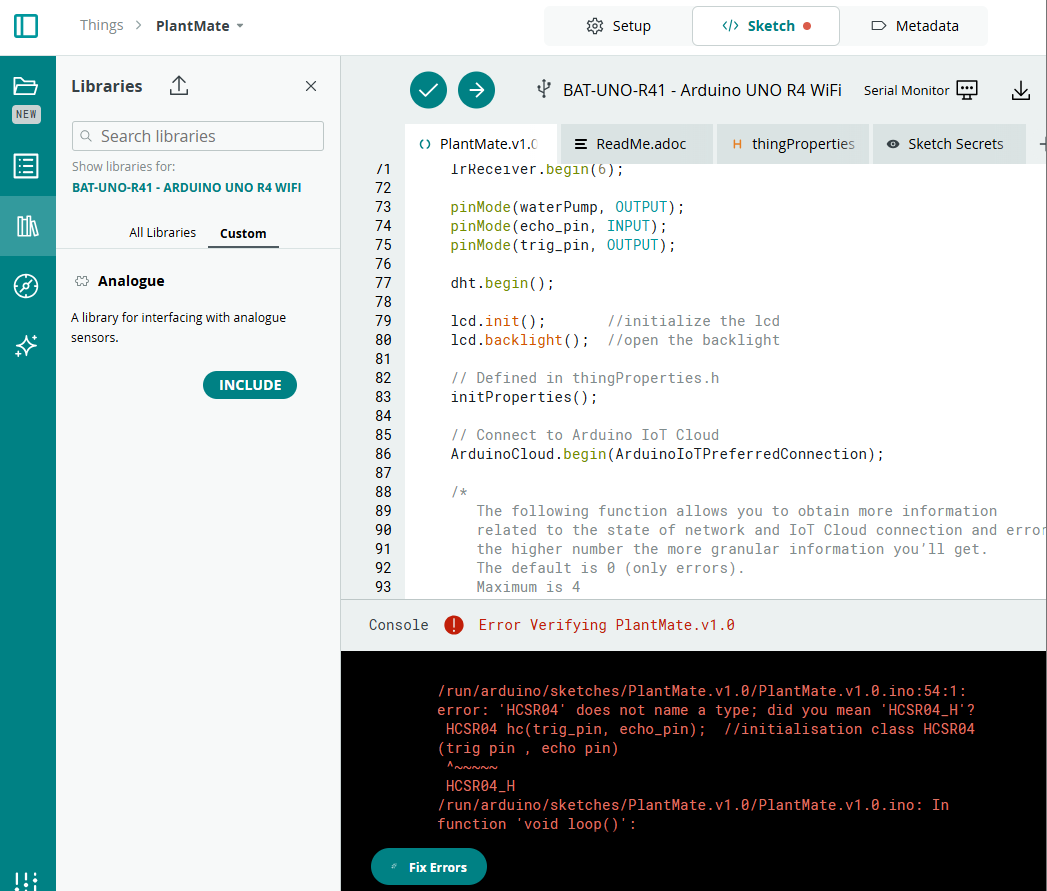
**Wi-Fi**

In the process of connecting PlantMate to internet via an existing WiFi network, I couldn’t connect PlantMate to my 5.0GHz home network. This was due to the Arduino IOT Cloud’s incompatibility talking to 5.0GHz networks. So I had to resort to connecting PlantMate to my 2.4GHz home network instead.

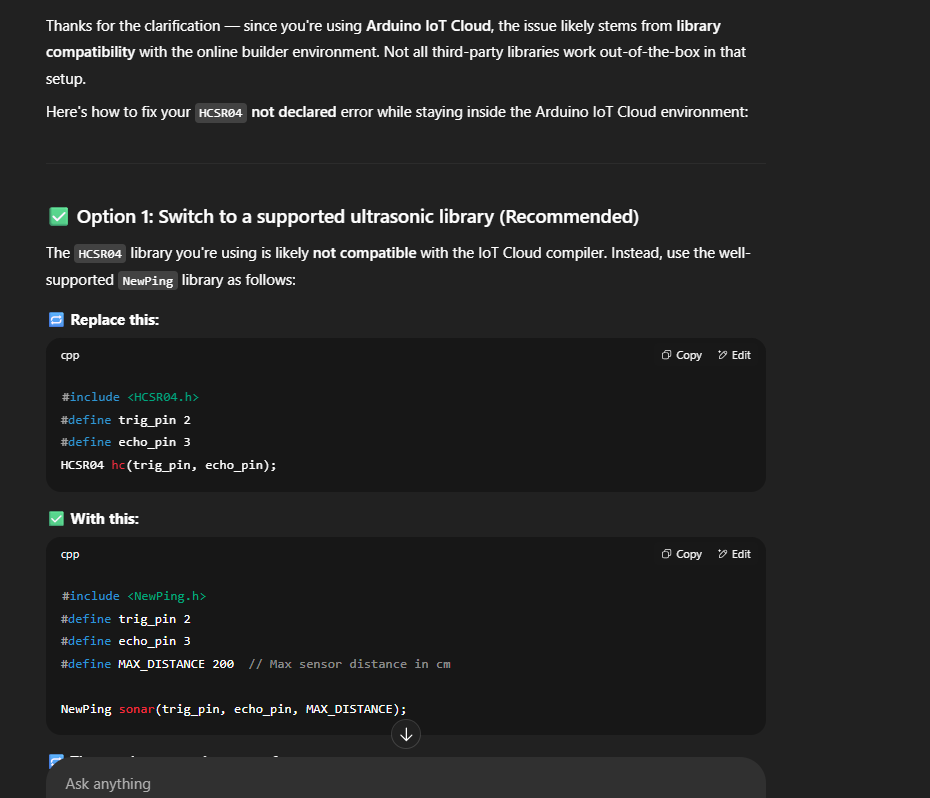
**Custom Libraries**

When uploading my custom library for interfacing with analogue sensors, there was something missing from the ZIP file containing the library. And that was a properties file called `library.properties`.

**Problem with HCSR04\_H Library on Arduino Cloud**

****

When trying to use the HCSR04\_H Library on the Arduino Cloud, it looks like the library was incompatible with the cloud. So I went to ChatGPT to find out what the problem was about.



ChatGPT acknowledged the problem of course being the Arduino Clouds builder environment not supporting libraries such as HCSR04, so It suggested I use a new library called ‘NewPing’.

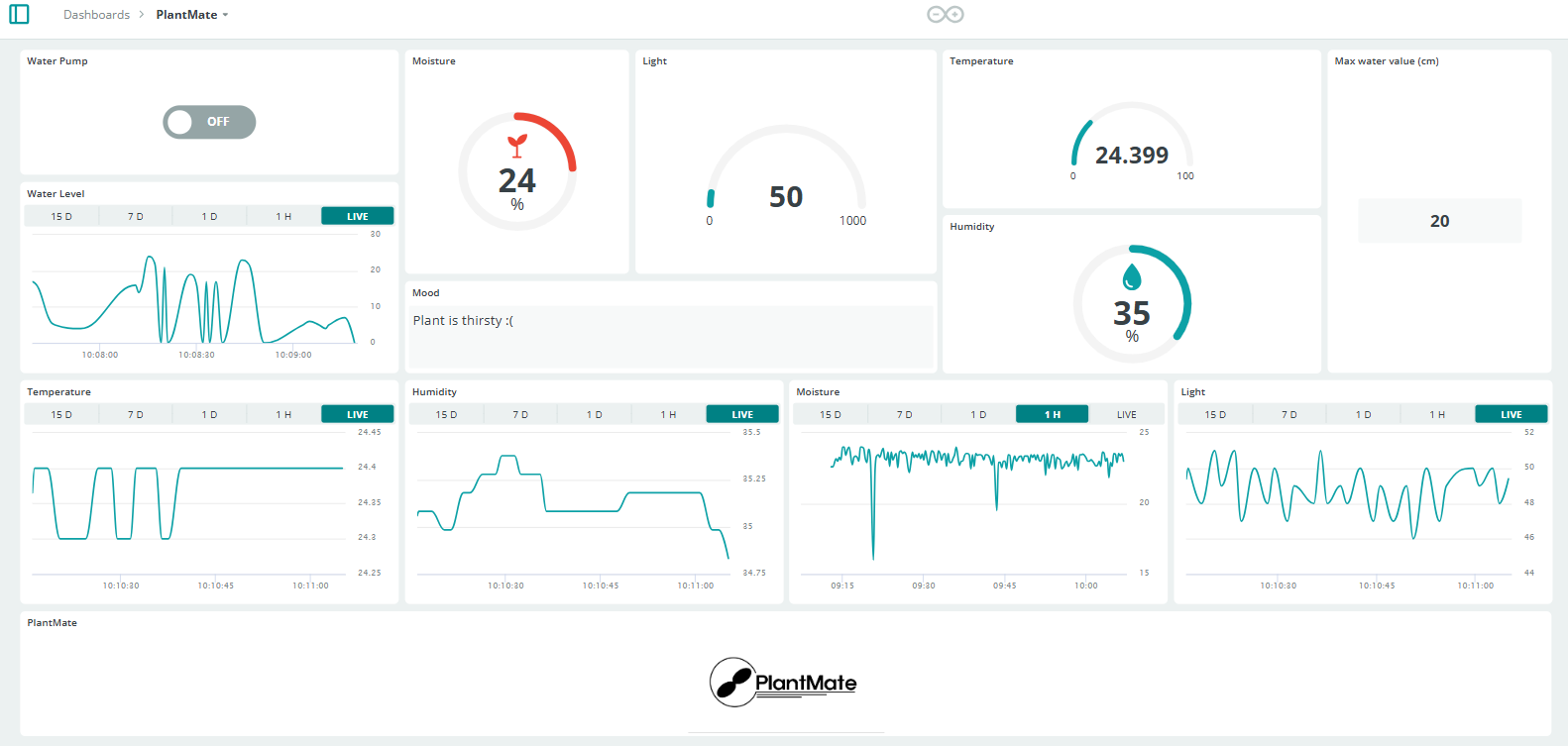
Next it suggested I replace all occurrences of the ‘.dist()’ method of the HCSR04 library with the `.ping\_cm()`method from the NewPing library.



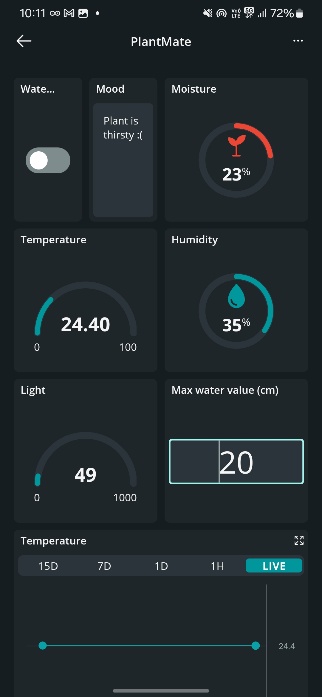
So not only did I add this new library to the online codebase for PlantMate, but on my local codebase too.

# Custom software

On Desktop:



On Mobile:



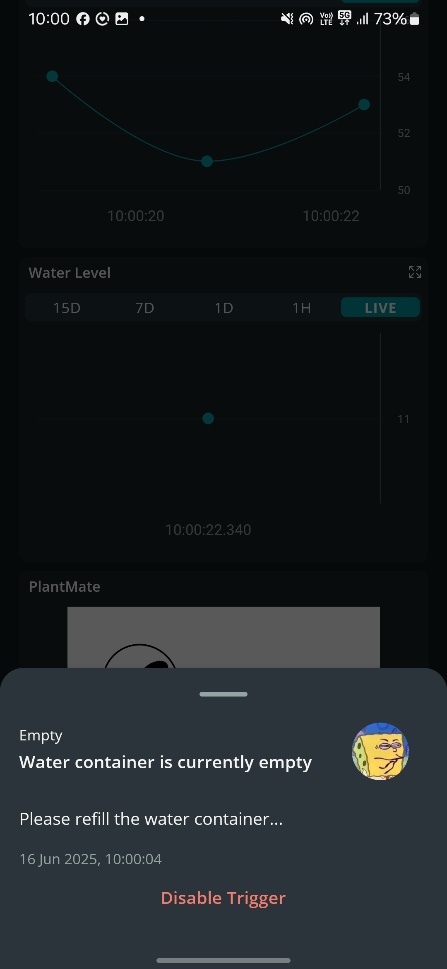
**Arduino IOT Cloud**

The custom software I am using for this project is the Arduino IOT Cloud. Thanks to this software I have been able to create a somewhat nice dashboard interface where the user can get a good overview of the plant’s current needs. At the top I have a switch where the user can manually turn the water pump on or off. And right of the switch are gauges and percentages displaying current soil moisture, light, temperature and humidity levels. At the bottom there are graphs where the user can view the trends across each data value.

**Push Notifications**

The latest feature I have implemented is where I receive push notifications based on various state triggers in the project, such as if the plant is thirsty, and if the pump is turned on.

A screen shot of a phone

AI-generated content may be incorrect. 

The third trigger is when the water container is empty, and my phone receives a push notification from that trigger.

# Future Scope.

The future scope for this project would be to more instances of PlantMate, basically have 3 to 4 or more small containers. This way I will be able to apply PlantMate as a better real-world product, without the size of the product being a big problem. Note it would also be good in the future to have a considerably smaller version of PlantMate as well.

One of my friends suggested possibly for the future, I could utilize a capacitive soil moisture module that contains a built in ESP32 and DHT11 sensor and runs on an 18650 battery. From this I gathered this will be very handy when applying PlantMate on a somewhat larger commercial scale.



If I do manage to have more of PlantMate, I would modify the current Arduino IOT Cloud web interface I have to monitor all these different instances of PlantMate. This way the user can monitor multiple plants all at once on unified web interface.

**Adjust plant needs based on species**

A possible feature in the future could be where the user can go to the PlantMate dashboard and enter in a plant species, and the values for water requirements, minimum and maximum soil moisture levels will be adjusted based on the specified species.

# Final Project.

A box with hoses on the floor

AI-generated content may be incorrect.



PlantMate has been constructed into a somewhat professional box setup, mainly to hide the spaghettified mess that is my wiring setup. The LCD screen is on the right-hand side of the front face of the box. The pump is on the left-hand side of the box. And last of all the moisture, light, infrared and temperature/humidity sensors are on the right-hand side.

**Remote-Control**

The user can leverage the use of remote-control technology to navigate through the PlantMate menu provided by the LCD screen. The user can view their plant’s current soil moisture levels, as well as temperature, humidity etc.

**Notifications**

The project also has functionality on the cloud where the user is alerted on their phone, about whether their plant is dry, the water pump is running and if the water container is empty. The user can also dynamically update the max water level on the PlantMate dashboard.

**Local instance of PlantMate**

If the user doesn’t want to use and interact with PlantMate via the cloud, they can utilize the code provided in the ‘Project’ directory of the PlantMate GitHub repo and upload that code via the Arduino IDE.

# Circuit Diagram.

