Autonomous Plant Watering System

Clayton Dsouza, *Student, University of Regina*

***Abstract*—Currently, gardening is considered as 1 of the most preferred hobbies and is likewise very significant for attaining environmental balance. Nonetheless, the main issue with gardening, is the time alteration, for irrigating the plant. Particularly, when caretakers are absent from their houses, for a couple of days, then the critical issue arises concerning the watering of the plants. In this modern high tech age, wherein people depend on technology for practically everything, in order to make their respective workloads less burdensome and more effective, a system designed to automatically water the garden containing the plants, without involving human interference, seems fitting. Thus, one of the key aims of my project, is to ensure that the physical activities of plant watering and tank refilling are replaced with an autonomous irrigation system and correspondingly excess use of water is minimized. The methodology deployed, continually senses the soil moisture levels, in order to determine whether or not, irrigation is required, furthermore, the amount of water required for the soil and the buffer tank, is determined as well. The program code concerning this project is based on Arduino UNO. This report articulates the proposed problem statement revolving around the essentiality of a circuitry that could automatically trigger a common DC water pump for the purpose of watering the soil associated with it. This process is programmed to occur in an event wherein the soil bearing the plant is nearly completely void of moisture. Correspondingly, my methodology toward undertaking the challenge of automated plant watering involved 8 crucial components. These components functioned interconnectedly to bring about the complete functionality of the system and comprised of A capacitive soil moisture sensing module, 2 DC water pump modules, 1 with an interconnected potentiometer module, an 8 (four Reds and four Greens) LED display module connected with a shift register, a water sensing module, a PIR sensing module with a corresponding LED notification system, a temperature sensing module and an LCD display module. The resultant functionality created the automatic plant watering system and is described as follows. On sensing an abnormal soil moisture level, the device will cause the water to be pumped from a buffer tank via the first DC motor, followed by the tank's replenishing, upon the water contained in it, going below a specific low point. Moreover, the temperature being constantly sensed is displayed along with the watering status and soil moisture levels, along with the presence of an obstacle, blocking the source of natural light. An option to manually override the watering system is also provided to the operator via a push button.**

***Index Terms*—Automatic Plant Watering, Soil Moisture Sensing, Water Level Sensing, DC Motor Pump with Arduino, Obstacle Detection with PIR Sensor**

I. Introduction

THE artificial methodology of watering soil bearing plants, presently being implemented, namely irrigation, is a widely implemented strategy for watering, especially in the regions experiencing a drier climate, where the occurrence of rainfall is scarce. This methodology has been proven successful, over the period of time, especially, in the aspect of suppressing the nourishment of unwanted weeds that grow and hinder the growth of the intended yield of crops, thus ensuring the apt growth of the plants therein. The former methodologies and practices implemented for irrigation were entirely non automatic and relied completely on physical labour, which they still require, to this day.

These techniques involved the utilisation of water containers and troughs, coupled with customised irrigation systems, the likes of which involve sprinkler irrigation system, a drip irrigation system and an area specific irrigation system, along with other lesser known techniques. However, when implementing these methods, the farmers or plant caretakers practically never take into consideration, the amount of water to be circulated through the watering channels and also the required level of water, needed to suffice the crops' water requirements. Attributed to this, the menacing issue of water logging, leading to the lessening of the nourishment value of the soil and also minor soil erosion take place. Hailing from a line of farmers, in India, I have witnessed such events of water logging, especially during the arid summer farming seasons.

Now, water preservation is the need of the hour, and these prevailing irrigation techniques, require a rigorous revamping, in order to ensure minimal water wastage, while watering the concerned plants optimally. Thus, in order to minimise the amount of water that is wasted through irrigation, I have developed an autonomous plant watering system, which also cuts down on the physical supervision essential to manoeuvre the system.

II. Problem Definition

In the current technologically advanced age, there has been a surge in the demand for strategies to automatically carry out the watering of plants, in order to tackle the menace of water logging and irregular plant watering routines. This phenomenon of water logging due to the irregular watering routines, be it for a house plant or for a vast farmland, has devastated the soil, by decreasing its nutritional value, thus adversely affecting the growth of the concerned plants. Therefore, one of the recently raised demands of the agricultural and irrigational sectors concerning various farmers and plant caretakers is an autonomous plant watering system.

Several published strategies for automating the plant watering systems document the implementation of a soil moisture sensing module with a corresponding motor pump, connected to a programmable logic board, such as the Arduino UNO, with some sort of notification system, to indicate the completion of the plant watering routine. These modifications are to be then integrated into the traditional irrigation systems, in order to witness a real world implementation of these strategies, on a larger and more effective scale. This is evidently visible, according to these statements from an article published on the smart irrigation market, "The increasing number of government initiatives to promote water conservation, the growth of smart cities, and the need for efficient irrigation systems, coupled with the reducing prices of controllers and sensors used in smart irrigation systems are chief driving factors that are surging the global market demand. For instance, according to the AsianInvestor, in 2021, there are above 500 smart-city projects are ongoing around the world with the technology spending of over USD 100 billion every year" [1].

Moving on, the autonomous plant watering system, documented in this report, mainly automates the watering of the plant and the buffer tank, upon detecting a lack of moisture content and water levels, respectively. The option for toggling the speed of the motor pump and turning it completely off as well, is also provided to the operator, in the form of a potentiometer. This was done, so as to further regulate the water flow, through the system, and prevent further wastage of water. Furthermore, a push button is integrated into the system so as to physically override the watering system and water the plants, as and when the operator chooses to. My approach to solving the problem of automated plant watering necessitates the integration of a capacitive soil moisture sensor, for sensing the soil moisture levels, a water level sensor with a corresponding LED Array display, to signify the amount of water in the buffer container, a temperature sensing module, an obstacle detecting PIR sensing module coupled with an LED for notification of an obstacle blocking the light source, an LCD display showing the current temperature and soil moisture levels, 2 DC motors that respond to the low water or moisture levels prompts concerning the buffer tank and the plant respectively, and the aforementioned push button.

Now, my approach successfully tackles the problem concerning the automation of the traditional plant watering routines, based on the variable soil moisture levels of the soil therein. However, there is tremendous scope for the further research and development of the hardware components involved along with a mobile software for monitoring the system's functioning.

III. Design Description

A. Overview

This report exhibits my approach toward solving the problem revolving around the essentiality of a circuitry that could automate the traditional irrigation routines, essentially, by manoeuvring the motor pumps on the basis of the integrated soil and moisture sensors, by turning them on for a specific duration, until the water requirements in both the potted plant as well as in the buffer tank are sufficed. This were to occur in the event wherein the soil has its moisture content below a specified lower limit and the buffer tank having a significant drop in its water levels. Subsequently, my approach comprised the 8 vital modules, for the purpose of producing a practical resolution to the problem of automated plant watering, each of which are defined in a comprehensive manner under the detailed description section.

These components functioned interconnectedly to bring about the complete functionality of the system and comprised of A capacitive soil moisture sensing module, 2 DC water pump modules, 1 with an interconnected potentiometer module, an 8 (four Reds and four Greens) LED display module connected with a shift register 74HC53N, a water level sensing module, a PIR sensing module with a corresponding LED notification system, a temperature sensing module and an LCD display module. The resistors coupled with the mechanisms as the 8 (4 Reds and 4 Greens) LED set and the implemented shift register, are of the value 10k ohms each. A common NPN BJT transistor is connected with the DC motor.

The resultant functionality created the automatic plant watering system and is described as follows. On sensing an irregular soil moisture level, the device will cause the water to be pumped from a buffer tank via the first DC motor, followed by the tank's replenishing, upon the water contained in it, going below a specific low point. Moreover, the temperature being constantly sensed is displayed along with the watering status and soil moisture levels, along with the presence of an obstacle, blocking the source of natural light. An option to physically override the watering system is also provided to the operator via a push button. The complete system is logically manipulated by the Arduino UNO, bearing the Atmega 328P chipset that controls the logic, by uploading of a C program to it.

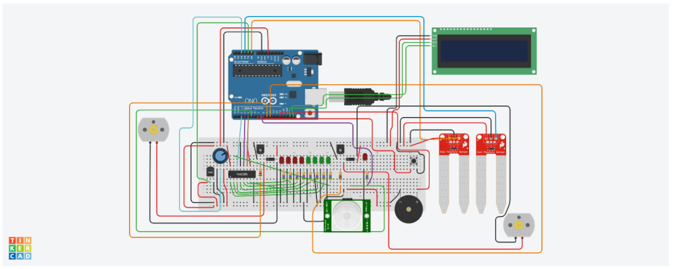

Fig. 1. Circuit View CAD depicting the complete circuitry

In the figure shown (Fig.1), the complete CAD schematic of the circuit comprehensive of the above mentioned modules as, temperature senor, shift register, soil moisture sensor, water sensor, PIR sensor, resitors, arduino, 2 DC motor pumps, 8 (4 Red and 4 Green)LED array, 2 NPN BJT transistors, LCD display 16 X 2 I2C , 2 Diodes and connecting wires are depicted.

B. Detailed Description

Corresponding to the proposed circuitry layout, the cohesive Components and apparatuses involved are as follows

- 1 Arduino Uno R3
- 2 DC Motor pumps
- 9 10k ohms resistors
- 2 550 ohms resistors
- 1 Push button
- 1 Temperature Sensor LM35
- 1 Shift Register 74HC595
- 1 250 k ohms potentiometer
- 1 16 X 2 LCD Display
- 4 Green LEDs
- 5 Red LEDs
- 2 NPN BJT transistors
- 1 PIR sensor

- 1 Water level sensor
- 1 Capacitive Soil Sensor
- 1 Piezo
- 2 Diodes
- Jumper wires

The inter connections amongst the various components as mentioned above, are documented as follows. Note, here, the Arduino Connections are referred to As Pins.

- DC Motor Pump 1 (positive) to BJT 1 Emitter
- DC Motor Pump 1 (negative) to Diode 1 Anode to Ground
- BJT 1 Base to 550 Ohms to Pin 9
- BJT 1 Collector to Cathode 1 of Diode
- Temperature Sensor LM35 terminal 0 to 5V
- Temperature Sensor LM35 terminal 1 to Pin A1
- Temperature Sensor LM35 terminal 2 to Ground
- 250 k ohms Potentiometer (wiper) to Pin A3
- 250 k ohms Potentiometer (positive) to positive
- 250 k ohms Potentiometer (negative) to negative
- Shift Register 74HC595 terminal VCC to 10k Ohms, 5V, MR
- Shift Register 74HC595 terminal DS to Pin 4
- Shift Register 74HC595 terminal OE to Ground
- Shift Register 74HC595 terminal STCP to Pin 3
- Shift Register 74HC595 terminal SHCP to Pin 2
- Shift Register 74HC595 terminal Q7 to Red LED 1 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q6 to Red LED 2 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q5 to Red LED 3 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q4 to Red LED 4 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q3 to Green LED 1 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q2 to Green LED 2 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q1 to Green LED 3 cathode of 8 (4 Red and 4 Green) LED array
- Shift Register 74HC595 terminal Q8 to Green LED 4 cathode of 8 (4 Red and 4 Green) LED array
- 8 (4 Red and 4 Green) LED array terminal Anodes to Ground
- DC Motor Pump 2 (positive) to BJT 2 Emitter
- DC Motor Pump 2 (negative) to Diode 2 Anode to Ground
- BJT 2 Base to 550 Ohms to Pin 9
- BJT 2 Collector to Cathode 2 of Diode
- 16 X 2 LCD (VCC) to positive
- 16 X 2 LCD (GND) to negative
- 16 X 2 LCD (SDA) to Pin SDA
- 16 X 2 LCD (SCL) to Pin SCL
- PIR (Data) to Pin 5
- PIR (positive) to positive
- PIR (negative) to negative
- Red LED 5 (cathode) to Pin 8
- Red LED 5 (anode) to negative
- push button (positive) to Pin 7
- push button (negative) to negative
- Capacitive Soil Sensor (Data) to Pin A0
- Capacitive Soil Sensor (positive) to positive
- Capacitive Soil Sensor (negative) to negative
- Water Level Sensor (Data) to Pin A2
- Water Level Sensor (positive) to positive
- Water Level Sensor (negative) to negative

The above connections can be visualised through the Schematic View of the components as in fig. below: The con-

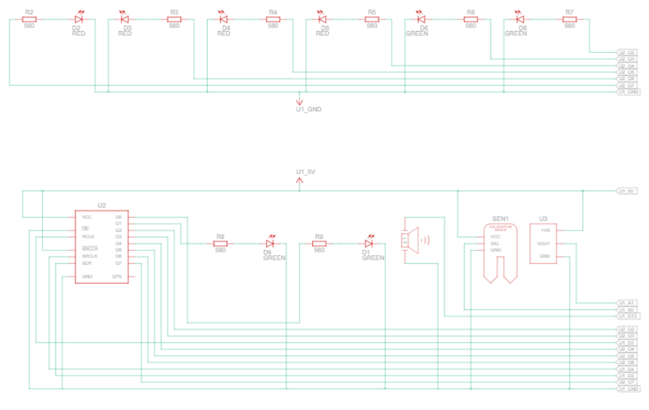
Fig. 2. Schematic View 1 depicting the complete circuitry

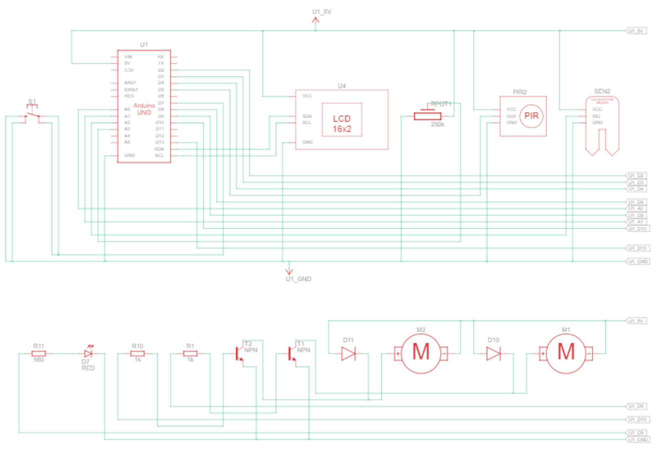
Fig. 3. Schematic View 2 depicting the complete circuitry

nections as carried out according to the detailed information above, can also be visualised in the pictorial formats, as seen in Fig. 2 and Fig. 3, highlighting the key aspects of the circuitry and charting out the entire connection map. These schematic maps go into immense details, to showcase the connections between each of the components detailed as above. Now, these components operate in unison, in order to attain the complete functionality of the system which essentially is to automate traditional irrigation and plant watering practices. These components when connected, come together to form 8 different modules, each of which play a role in the plant watering process, with the exception of the temperature sensing module, that merely monitors the room temperature in the vicinity surrounding the concerned plants. Furthermore, although the

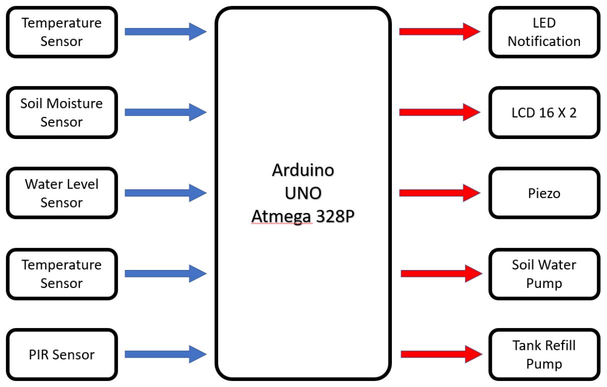

Fig. 4. Block diagram depicting the complete circuitry

temperature does impact the growth of the plant, in terms of varying the rate at which the plant would run out of moisture from the soil it thrives in, I haven't programmed the Arduino to respond to this stimulus. Thus, the 8 different modules are as follows, the temperature sensing module, the water level sensing module, the soil moisture sensing module, the plant watering module, the tank refilling module, the obstacle sensing module, the manual overriding module, and lastly the display and notification modules, encompassing the single Red LED, the 8 LED array paired with the Shift register and the LCD display. Starting with the temperature sensing module; As mentioned earlier, this particular module merely reads the temperature continually in an analogous manner and feeds it into the Arduino. Theoretically, the LM35 is a dynamic IC Temperature sensor, in which its produced voltage oscillates, according to the room temperature surrounding it. Furthermore, it is fundamentally a miniscule and budget friendly IC that could be used for the purpose of computing temperature within the vast range of -58°C to 147°C. Moreover, the real time, analogously sensed temperature values are printed on the 16 X 2 LCD display connected into the circuitry. Although this mechanism is not a vital component in tackling the problem statement, integrating this module into the circuitry could assist the operators in perceiving the surrounding temperature without having to utilise a peripheral temperature reading device.

Next, the water sensing module is integrated into the circuit as, it performs one of the core functions of the entire circuitry. This function is in keeping with the water contained in the buffer container. For further comprehension, the buffer container acts as an intermediary water holding container, between the potted plant and the main overhead tank or water resource. The buffer container serves the purpose of being a backup water resource, in the event of a water outage or a main tank cleaning process, thus ensuring that, the concerned plants always have sufficient water. The water level sensor, is strategically placed in the buffer container at a specific point along the edge. Now, the original design utilises an ultrasonic sensor for the purpose of water level measuring, however, these are easily manipulatable as, if an object were to fall on the ultrasonic sensor, like a leaf, flower or any such item, the ultrasonic sensor would detect a false spike in the water levels. On a large scale, where such events are more likely to occur such as on a farmland, these erroneous false water level spikes could appear more often and cause the circuit to malfunction in this regard. Thus, the decision of using a water level sensor instead was made, as the water level sensor accurately predicts the amount of water present in the container. Now, corresponding to each level on the water level sensor, a grading scale is computed within the range of 0 to 600 and above, wherein, the water level ranging from 0 to 509 indicates that the buffer tank needs to be refilled and the tank refilling module is subsequently triggered. The values from 510 to 549 indicate that the tank contains low water levels; At reaching this level, the tank refilling module stops pumping water. The values from 550 up to 600 indicate that the tank is at a medium water level. The values beyond 600 indicate that the tank is full. These labels, viz. the low, medium and high labels are visualised in terms of the glowing and dimming of the LEDs within the 8 LED array. When the tank requires a refilling, only the first red LED would glow, followed by the subsequent 2 LEDs glowing as the tank gradually starts to fill up at the Low mark. As the tank continues to fill up, the next 2 LEDs would glow indicating that the tank is now at medium level. Lastly, all 8 LEDs would light up, if the buffer tank is completely filled up or it crosses the medium grade point. The interlinked tank refilling module is programmed to run for 5 seconds at a stretch, each time it pumps water into the tank.

This is followed by the inclusion of yet another core component, that being the soil moisture sensing module. The soil moisture sensor detects the amount of moisture present within the soil, in an analogous fashion, in real time. Here, a constant threshold value is defined as the dryness point. In the event that the soil was to nearly completely run out of moisture, the sensor would detect a value beyond the dryness constant. At that very instant, the Arduino prompts the soil watering module to start. As the moisture content in the soil increases, the soil watering module stops pumping water into the potted plant. The interlinked soil watering module has an integrated potentiometer, so as to regulate the flow of water. This watering module has an integrated piezo as well, to further notify the user of the completion of a watering cycle of 5 seconds. The automated watering process continues to take place up until the soil moisture levels cross the defined dryness constant. Now, an overriding module is present as well, so as to allow the operator to manually conduct the watering of the plant, for a stretch of 5 seconds, each time they click the push button. The LCD display is interconnected so as to display the status of the soil moisture levels, when the watering is taking place.

The obstacle sensing module is interlinked with the single Red LED notification module, so as to detect the presence of any object, obstructing the plant from receiving light from a natural or artificial source. The PIR sensor is used in this case and detects changes in the quantity of infrared radiation incident upon it. According to the PIR page on Wikipedia, "Whenever an object, such as an individual, steps in front of the background, like a wall, the corresponding temperature at that specific region in the PIR sensor's viewing spectrum would increase from room temperature to body temperature, and then plumet down again. The sensor translates the subsequent variation in the inbound infrared radiation into an alteration in the yielded voltage, and this event causes the detection to occur". In our case, however, we would be placing the PIR sensor in front of the light source. Upon detection of such an obstacle, the Red LED would glow, for as long as the obstacle were to be present in front of the PIR sensor and move.

In the figure shown (Fig.4), a block diagram is created, as an overview, underlining the operation flow as mentioned above, by means of inputs and outputs, in which, the components such as the Temperature Sensor, water level sensor, soil moisture sensor, push button and the potentiometer are the input modules to the processing Unit, namely, the Atmega 328P on the Arduino Uno, which mirrors as the yielded outcome to the 2 DC Motor pumps, The Shift Register impacting the 8 (4 Red and 4 Green) LED array, the piezo, the Red LED and the 16 X 2 LCD Display.

C. Use

The operational process concerning the system corresponds to the utility manual as documented below. This utility guide was drafted according to the prototypical product of the autonomous plant watering system that is to be deployed on a small scale, typically for potted plants only. The prototype mentioned above, is illustrated in Fig. 5, with a sketch version in Fig. 6 for further comprehension of the system's operations. Moreover, the plant caretaker is to strictly follow the guide pointers as listed below, and not fiddle around with the internal circuitry, since this could probably cause the device to malfunction and thus not fulfil its intended purpose. Note, in the prototype, there would be several wires floating about the device. None of the wires are to be disconnected under any circumstances. These are the steps that the operator must follow, in order to get the device to function as programmed.

1. Connect the various components of the system, in the manner as pictorially represented in the sketch in Fig.6. Note, have 2 water resources, a smaller buffer container wherein you should insert the soil watering motor pump (connected with the green wire) and the water sensor as well. Another larger container or an external tank is to be used as the secondary water source, wherein you should place the tank refilling motor pump (connected with the purple wire). The potted plant is to be placed in close vicinity of the setup box, with the soil moisture sensor and the soil watering pipe fitted in the soil.

2. Connect the automated plant watering system to an AC power outlet and verify if the integrated 16 X 2 LCD display has commenced showing the temperature values and the soil moisture values subsequently.

3. Now, at first, if the soil moisture sensor is not inserted into the soil, or if the soil surrounding it is dry, the same should reflect on the LCD screen. Check if the LCD display is showing a value lesser than the programmed dryness value, 270, as the current Moisture level. In such a scenario, verify if the soil watering motor pump has started pumping water.

4. Check the buffer tank water levels and verify if these levels correspond with the LED display array's lights. If the tank is empty, check to see whether the tank refilling pump has started pumping water from the secondary water resource.

5. Now, check to see whether the Soil water pump varies its speed upon turning the potentiometer both clock wise and anticlockwise. Furthermore, Upon the soil water pump completing its automated watering cycle, listen for a melody from the integrated speaker.

6. Check if the manual overriding option is functioning by clicking the blue push button. Upon clicking it, the pump should pump water for 5 seconds and then stop. Here, however, no melody would be played, as, you are manually overriding the system.

7. Check to see if the motion sensor, is functioning as intended, by placing it near the supposed light source, from which the plant would carry out its photosynthesis routine. Place an object in front of the PIR sensor and verify if the red LED starts glowing, indicating the presence of an obstacle.

IV. EVALUATION

A. Overview

This section focuses on the comprehensive photographic evidence of the circuitry developed in accordance with the documented problem statement. The figure depicted in Fig.5, shows the detailed prototype along with the sketch version in Fig.6, that could be further enhanced and developed into a fully functional model, upon overcoming the limits of the prototype. These limitations, pertain to the hardware limitations and breakdown, which occurred unfortunately, during the testing phase, and are articulated under the assessment section. However, the core functionality of the plant being watered adequately and efficiently was fully automated, in the prototype, thus solving the problem of automated plant watering or irrigation. Now, the entire circuitry was first designed in Tinker CAD and successfully simulated, in order to understand the manner in which the circuit would function in real world, prior to performing the physical circuit connections. Upon successfully connecting and simulating the circuitry on Tinker CAD, the Schematic View and Circuit View diagrams as seen in Fig 2 and Fig 3 respectively were used as references throughout the physical connection procedure. Now in the simulation, the tests involving the varying of the temperature sensor, the soil moisture sensor, the water level sensor and the PIR sensor were carried out along with the clicking of the push button, for the manual overriding process as aforementioned. Furthermore, the notification and display modules were also checked with regards to the manipulation of the input sensors' values as mentioned above. Further details on the manner in which these tests were conducted could be seen under the testing and results section, with the results documented under the assessment section.

My tactic utilised to construct and deploy the circuitry as a solution to the problem statement was straightforward and was performed after procuring the various components and apparatuses, vital in accomplishing the same. According to this tactic, the operator is provided with the ability of viewing the real world conditions concerning the potted plant such as the surrounding temperature, the soil moisture levels and the water tank levels on the LCD display and is also notified about the completion of the automated watering cycle, via the piezo, and about the presence of an obstacle via the glowing Red LED. Now, the operator has the option of manually overriding the soil watering cycle as well as controlling the rate at which the DC pump would pump water into the soil. Whenever the soil moisture level drops below 510 and the water sensor values would cross the dryness point, the soil water pump and the tank refilling pump, respectively, would start pumping water, from their connected water sources.

Thus, I developed only one prototype, that went beyond solving the problem of automated plant watering and irrigation, through the integration of more sensors, the displays and an additional motor pump, as feature enhancements. The components required for the entire process could be found under the detailed description section. Furthermore, this project was sketched, simulated, developed, tested and articulated in a physical form, all by myself.

B. Prototype

The prototype used and described in this section was implemented as the final version as well. This prototype is graphically represented in the form of the schematic views and circuit views as seen above, in figures, fig.1, fig.2, and fig.3 respectively. The entire prototype was furnished in accordance with the pictorial representation of the connections as seen in the CAD format, as in fig. 1, as an exact replica of the circuitry portrayed. This prototype can be seen in fig. 5 as a photographic image, as evidence of the physical construction of the circuitry and in fig.6 as a sketch of how the prototype were to look, if deployed on a real world potted plant as a finished product. Now, more images showing the various components connected physically, are included in the Appendix section, for further reference. The connections between each of the various components involved in the circuitry are listed out in a precise fashion, under the detailed description section. The original version of this project can be found on the Arduino cc webpage and is also included in the form of a weblink under the references section. My approach toward the construction of this specific prototype was solely based on inspiration and creative enhancement at every milestone. I had 3 distinct milestones that I'd tasked myself with accomplishing, during the course of building the prototype. These milestones were practically easy to achieve, with the exception of a component failure. However, each milestone, superseded the previous, in terms of the addition of more features, thus enhancing the prototype as a whole, from the original version as aforementioned. Following are the 3 milestones that were chartered out.

Firstly, I simulated and developed the entire prototype, in accordance with the original version of this project as mentioned above. Here, I was able to successfully tackle the problem of automating the irrigation or plant watering routines, and furthermore collect data from the surrounding environment such as the soil moisture details, the tank water levels and the temperature in the vicinity of the potted plant. Here, I was able to successfully alert the operator about the tank water levels, and the completion of the automated soil watering routine, wherein, the soil watering took place, in an accurate manner, whenever the Arduino read in the triggering values from the corresponding soil moisture sensor. Furthermore, the operator would also be notified in the event that an obstacle were to be blocking the source of natural or artificial light. However, a slight setback occurred at this step, as although the LCD display was showing the temperature during the testing phase, it suffered a critical damage to its I2C board during the trials, which caused it to display gibberish values. This setback however, did not deter me from pursuing my goal of enhancing the technical features of this project, mainly due to the fact that my main goals of automating the traditional plant watering system and detecting the water container levels were achieved. These factors gave me the motivation I needed to pursue my next 2 milestones.

My next milestone was to develop a mechanism, so as to refill the tank from which the water was being pumped from, in order to water the potted plant. This was done, in order to have a backup resource of water, in the event that the main tank was to undergo a tank cleaning process or were to be shut down for certain other reasons. In such cases, a backup tank was to draw its water from the main tank, in the event that the backup was to have its water level below a specific programmed constant value. Now, the refilling process involved the tank water sensing module as well as the tank refilling module, in the methodology as described in the detailed description section. This was tested, for the accuracy in terms of when the actual tank refilling process would commence, by repeatedly inserting and drawing out the water level sensor at different graded standpoints, in the backup tank. Upon attaining success in this milestone, I was further motivated to pursue my last milestone in the construction of the prototype.

My last milestone involved the feature enhancements concerning the soil watering module. In all its essence, the soil watering module was successfully watering the soil, whenever the soil moisture sensing module would read in a value lower than a specified programmed threshold limit. However, I visualised 2 distinct scenarios wherein, merely achieving the automation of the soil watering routine, wouldn't be enough. The first scenario involved the event wherein the operator would want to customise the rate at which the pump would pump water into the soil. This motivated me to integrate a potentiometer into the circuitry that would allow the user to manoeuvre the rate at which the water was pumped into the soil. This would further enhance the water conservation features of this project. The next scenario involved the event wherein the operator found the system to be malfunctioning, with regards to not watering the plant upon detecting dry soil. Although the occurrence of such an event is quite rare, the lifetime of the capacitive soil sensors is known to be very short, moreover, these sensors are known to accurately sense the soil moisture levels' variation, however, along a fairly inaccurate scale. This motivated me to integrate a push button into the circuitry, to manually carry out the soil watering routine, by overriding the programmed logic and having the soil watering module pump water into the soil, for a duration of 5 seconds. See Fig.5 for a pictorial representation of this prototype.

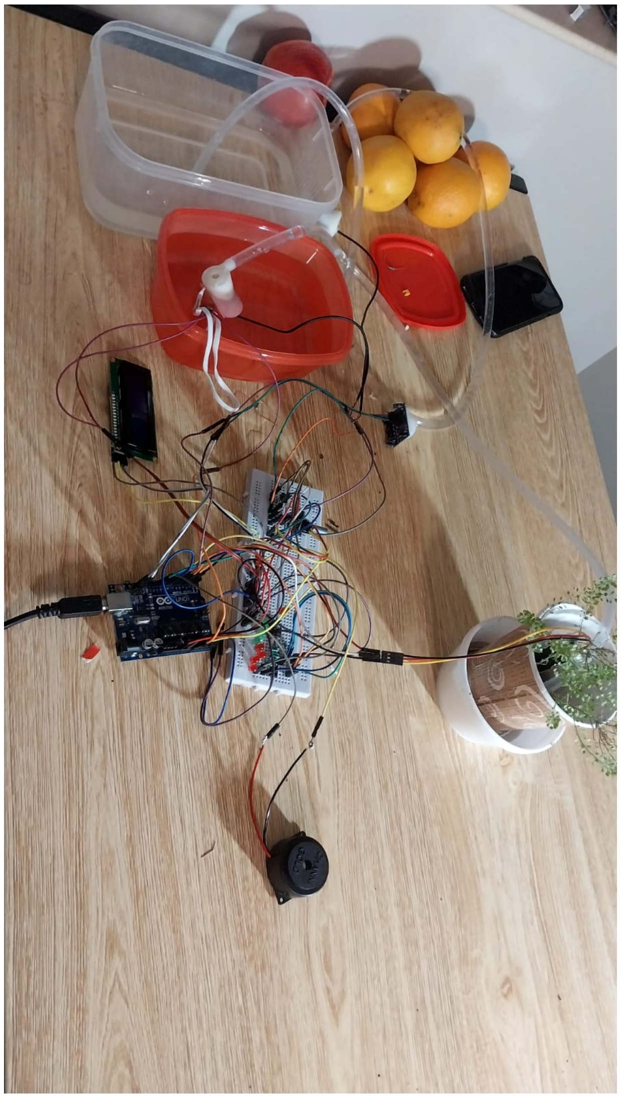
Fig. 5. Pictorial representation of prototype

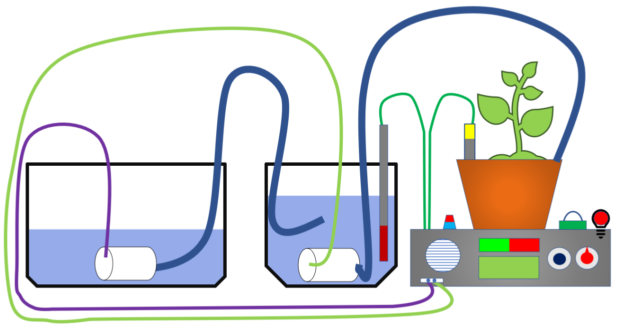
Fig. 6. Sketch View depicting the prototype

C. Testing and Results

The following tests were conducted by me, in order to verify the accuracy of the constructed prototype. These tests, were mainly to check, how different real world scenarios would trigger the Arduino to respond, which further relied on the accuracy of the values being read in by the different connected sensors such as the soil moisture sensor, the water level sensor, the temperature sensor, the obstacle detection sensor and the push button. The output devices such as the 2 DC water pumps, encompassing the soil watering module and the tank refilling module respectively, the 8 LED array integrated with the shift register, the Red LED notification LED, the piezo and the LCD display were checked, in order to validate the outputs and the accuracy of the tests conducted. Thus, a total of 4 different tests were conducted based on the relevant parameters for each test which are described as follows. Note, however, in all of the tests conducted, the LCD 16 X 2 test was eliminated, as it did not make it through the prototyping phase, since it incurred a damage during the second milestone of the development phase.

Firstly, the soil moisture sensor along with the soil watering module was tested. The soil moisture sensor was tested by gradually dipping and removing the sensor from the soil, under different soil moisture conditions. These conditions included, a totally dry soil, which definitely needed water, a slightly moist soil, wherein watering was not mandatory and a fully moist soil wherein the soil did not require any further watering. Correspondingly, the soil watering pump was tested concerning its response to the various stimuli as mentioned above.

Next, the tank refilling module was tested along with the tank water level sensing module. This test was performed, by varying the position of the water level sensor, by drawing it from and inserting it deeper into the water at various intervals. These intervals corresponded to 4 distinct programmed benchmarks, at which the interlinked 8 LED array display would glow the specified LEDs. These benchmarks were labelled empty, low, medium and high, wherein, the LEDs would either glow the only the first, or the first 3 or the first 5 or all the LEDs at once, respectively. The tank refilling pump was also tested to verify its watering routine, when the water level sensor detected that the buffer tank was void of water.

Next, the obstacle detection module was tested to see if it accurately detected the presence of an obstacle in its sighting region. Here, the PIR sensor was first strategically placed in a region of the natural or artificial light source. This was followed by the placing of different objects in front of the PIR sensor, to check if the interlinked LED would glow, each time this occurred.

Lastly, the manual overriding module along with the potentiometer interlinked with the soil watering pump were tested. These tests involved clicking the push button, to check whether the soil watering pump turned on for a duration of 5 seconds. The next test concerning the potentiometer, was performed to check whether, upon rotating the potentiometer clockwise or anticlockwise, the soil watering pump was increasing or decreasing its rate of dispensing the water into the potted plant's soil, respectively.

The results of each of the above tests, as to how the prototype developed by myself, stood up to each of these tests, are documented under the assessment section.

D. Assessment

The tests as mentioned above were conducted by myself, upon completing the prototype build, according to the problem statement and the enhancements documented as milestones under the prototyping section. These tests were performed, bearing in mind the real world conditions that the automated watering system would be faced with. The results of said tests are documented as follows, in order to show how the constructed prototype stood up to the tests. Note however, he 16 X 2 LCD display crashed critically during the second milestone along the prototype building phase. This setback was due to a damage that occurred with the I2C chip, causing the LCD to display gibberish values. The rest of the component however successfully passed each of the tests, when subjected to the various scenarios as documented in the test and results section.

Firstly, the soil moisture sensing module was tested, to check whether or not, the soil moisture sensor was detecting the soil moisture levels accurately. This was done by repeatedly inserting the soil moisture sensor into the pot at various intervals, each interval being two hours apart from the prior interval. The soil moisture sensor, successfully sensed the plumet in the coil moisture levels, when it was completely taken off the soil, at nearly zero moisture levels. The sensor successfully detected when the soil moisture level dropped below a specific low point as well, triggering the Arduino to start the soil watering pump for a duration of 5 seconds. This automated watering routine was demonstrated effectively along with the piezo that played a wonderful melody at the end of the watering routine.

Next, the tank refilling module was tested to see whether the tank would get refilled by the tank refilling module automatically. The tests conducted involved inserting and taking off the water level sensor from the container at various grade points, demarcating the empty, low, medium and high, water levels in the tank, according to the programmed software. The conducted test yielded 2 successful outcomes. Firstly, upon completely drawing out the water level sensor from the buffer tank, the tank refilling process commenced, with the tank refilling motor turning on for a stretch of 5 seconds. Next, the interlinked LED array display successfully lit up the concerned number of consequent LEDs, in order to indicate the amount of water present in the buffer water tank at each testing interval.

The obstacle detection module was then tested to check if the sensor was accurately detecting the presence of an obstacle, blocking the source of natural or artificial light, essential for the plant's growth. This module successfully detected the presence of an obstacle whenever, an object was moved in front of the PIR sensor. Now, as per the test described concerning this module, an LED had to glow, in the event that an obstacle was detected. The glowing of the LED was achieved, as an indicator, to notify the operator about the presence of the obstacle.

The last 2 tests revolved around the functionality of the soil watering module. The first test here, concerned the manual overriding of the soil watering module. The reason behind integrating this module is documented under the prototype subsection. However, the test results concerning the pushbutton revealed that the push button was functioning as intended, with each click, triggering the turning on of the soil water pump for a duration of 5 seconds. The next test, subsequently, involved the variation of the rate at which the water was being dispensed from the soil watering pump. Upon turning the potentiometer clockwise, the soil watering pump was observed to be dispensing water at a faster rate, whereas, on turning the potentiometer anticlockwise, the soil watering pump was observed to be dispensing water at a slower rate.

Now, with the exception of the LCD display module, due to the damages it incurred, as mentioned above, all the rest of the components functioned as intended.The LCD could not be replaced as, procuring its replacement was too expensive, at the time of documenting this report.

E. Next Steps

In order to be employed as a real-world application to support in the Automated plant watering experience, the current prototype would have to undergo thorough hardware and software modifications. In this project, the problem statement documented has, at the time of documenting this report, been fruitfully attempted and resolved with regards to the prototype's working, except for the malfunctioning LCD 16 X 2. Therefore, following are the enhancements that should be accomplished in order to advance the system, physically, and attain full user compatibility. As the LCD is of vital importance to the system's functionality, it must be procured immediately and handled with extreme precautions, while tinkering with it. Secondly, this version of the prototype features a detailed complex wired circuitry. However, adding a wireless component such as a Bluetooth module would further enhance this project, as in accordance with the stretch goal documented in the project proposal. Here, the Bluetooth sensor must be integrated with the current system prototype, in order to broadcast the sensor values from the Arduino to an Android phone application. This application could also perform a 2 way communication, with the system, by controlling the manual overriding soil watering module, from the application itself. Lastly, the entire system, as seen in fig. 5 must be encased in a 3 D printed chassis, as seen in the sketch version of the prototype, in fig.6. This would ensure the safety of the components and the logic board from any form of physical damage, especially water damage. These enhancements, unfortunately, couldn't be carried out by myself, due to the lack of the hardware and computational resource.

V. Conclusion

In conclusion, the solution proposed by myself, in correspondence with the prototype I developed, in response to the problem of automated plant watering, were demonstrated both through simulations and a functioning prototype. These demonstrations were carried out, in the from of a series of tests, the details of which are included under the Tests and Results, and the Assessment sections. The prototype, achieved the primary goal of successfully automating the plant watering routine, however, a waning LCD 16 X 2 hindered it from becoming a fully functioning prototype as programmed. All sensors and actuators, apart from this mishap, were fully functional, in coordination with the programmed software uploaded to the Arduino UNO. Upon succeeding in the prototype building milestones, I included certain enhancements that could be done in the future under the next steps section, as a head start for the developers working on this project in the near future.

Appendix A
Diagrams and Pictures

Any extra diagrams and pictures go here.

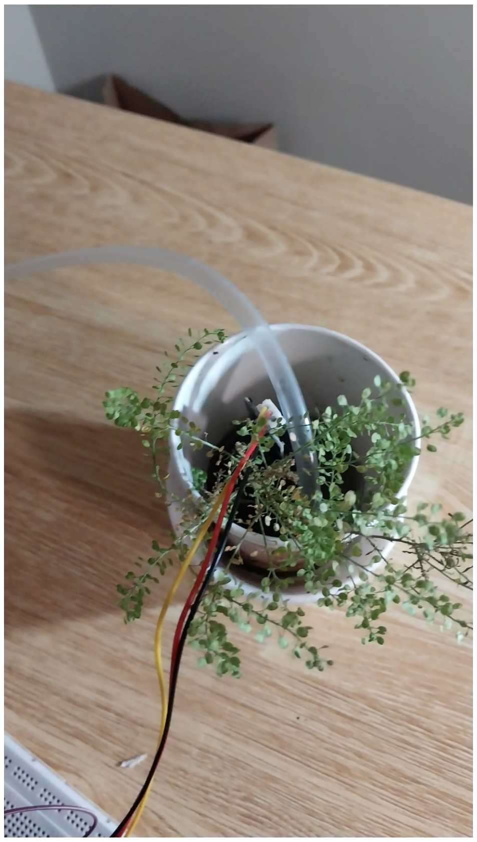

Fig. 7. Capacitive Soil Sensor View

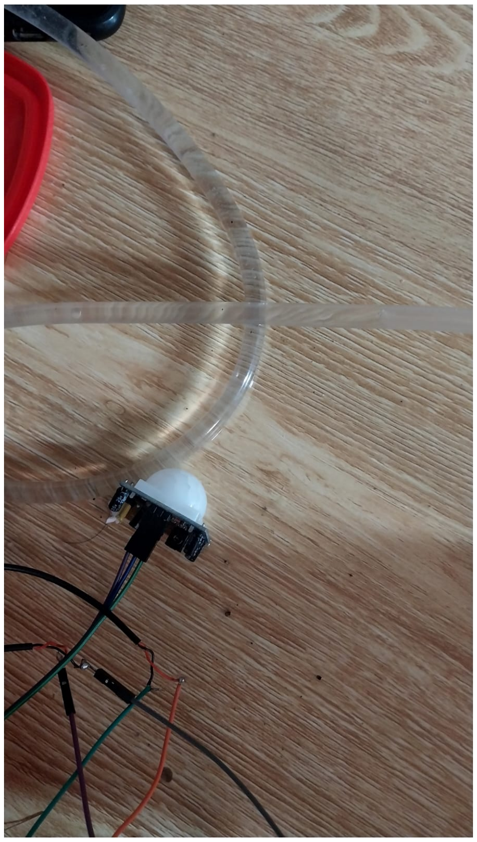

Fig. 8. Obstacle Detection PIR Sensor View

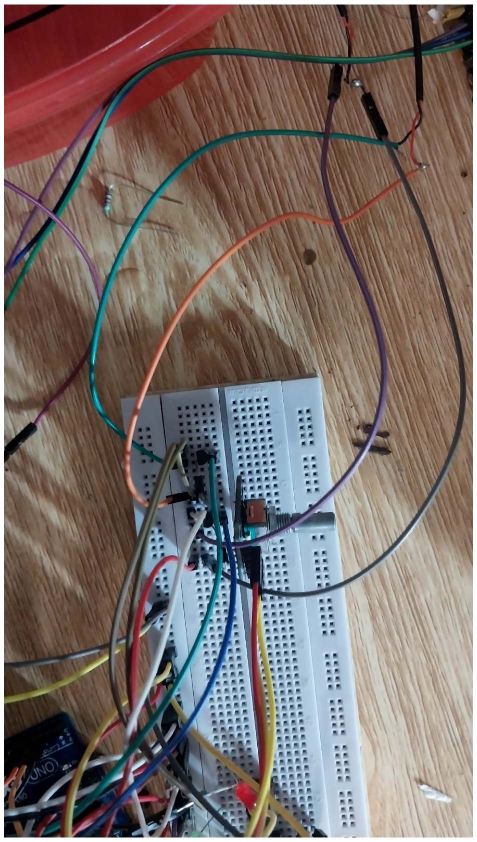

Fig. 9. Soil Water Motor Pump Potentiometer View

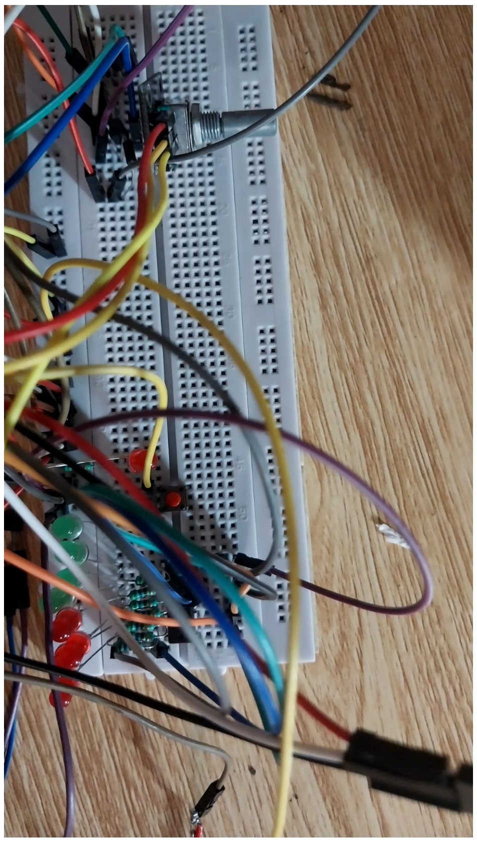

Fig. 10. Prototype Manual Override and LED Array View

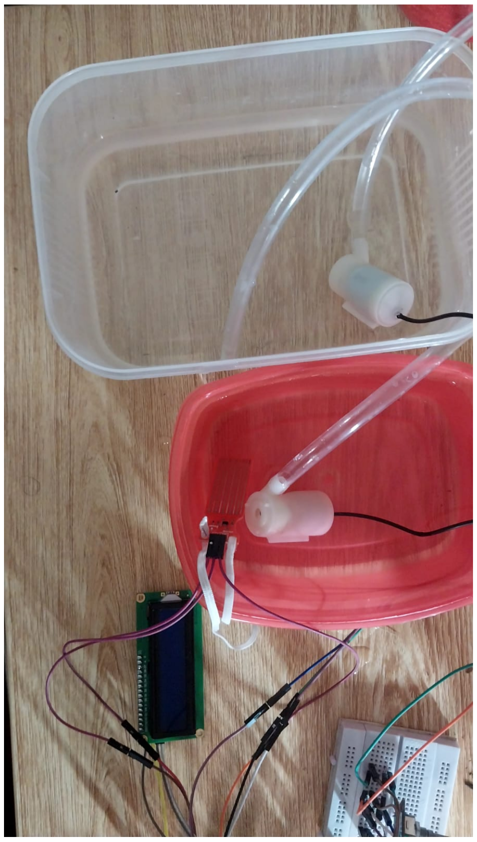

Fig. 11. Motor Pumps and LCD Display View

APPENDIX B
SOURCE CODE

The source code that YOU WROTE goes here. If you didn't use your own code, you need to reference that fact in the design description.

```
#include <Wire.h>
#include <Adafruit_LiquidCrystal.h>
Adafruit_LiquidCrystal lcd_1(0);
#define LATCH_PIN 3 // CI 12
#define CLOCK_PIN 4 // CI 11
#define DATA_PIN 2  // CI 14
//Music
#define BEAT 300
#define PIN 13

#define DO 262
#define RE 294
#define MI 330
#define FA 349
#define SO 392
#define RA 440
#define SI 494
#define HDO 523

//Timer
//int counter = 0;
int water = 0;
const int dry = 270;
const int pumpPin = 9;
const int tankPin = 10;
const int soilSensor = A0;
const int waterSensor = A2;
const int poten = A3;
const int pirSensor = 5;
const int BUTTON_PIN = 7;
const int obstacle = 8;
int buttonState = 0;
int var;
int hiz = 0;

void setup() {
  pinMode(pumpPin, OUTPUT);
  pinMode(tankPin, OUTPUT);
  pinMode(obstacle, OUTPUT);
  pinMode(soilSensor, INPUT);
  pinMode(waterSensor, INPUT);
  pinMode(pirSensor, INPUT);
  Serial.begin(9600);

  lcd_1.begin(16, 2);
  pinMode(BUTTON_PIN, INPUT_PULLUP);
}

void loop() {
  int sensorval = digitalRead(pirSensor);
  Serial.println(sensorval);

  if (sensorval == HIGH) {
    digitalWrite(obstacle, HIGH);
    delay(50);
    digitalWrite(obstacle, LOW);
    delay(50);
  }
  else {
    digitalWrite(obstacle, LOW);
  }
  ckeckBME280();
  checkWater();
  checkMoisture();
  buttonState = digitalRead(BUTTON_PIN);
  if(buttonState == LOW)
    digitalWrite(pumpPin, HIGH);
  int sensorvalue = analogRead(A3);
  int voltage = sensorvalue * (10 / 1023.0);
  //Serial.println(voltage);

  switch(voltage)
  {
  case 1:
    Serial.println("State 1 ");
    analogWrite(pumpPin,0);
    break;
  case 2:
    Serial.println("State 2 ");
    analogWrite(pumpPin,voltage);
    break;
  case 3:
    Serial.println("State 3 ");
    analogWrite(pumpPin,voltage);
    break;
  case 4:
    Serial.println("State 4 ");
    analogWrite(pumpPin,voltage);
    break;
  case 5:
    Serial.println("State 5 ");
    analogWrite(pumpPin,voltage);
    break;
  case 6:
    Serial.println("State 6 ");
    analogWrite(pumpPin,voltage);
    break;
  case 7:
    Serial.println("State 7 ");
    analogWrite(pumpPin,voltage);
    break;
  case 8:
    Serial.println("State 8 ");
    analogWrite(pumpPin,voltage);
    break;
  case 9:
    Serial.println("State 9 ");
    analogWrite(pumpPin,voltage);
    break;
  case 10:
    Serial.println("State 10 ");
    analogWrite(pumpPin,voltage);
    break;

  }
}

//Temperature
void ckeckBME280(){
  int tmp = analogRead(A1);
  float voltage = (tmp * 5.0)/1024;
  float milliVolt = voltage * 1000;
  float tmpCel =  (milliVolt-500)/10 ;
  float tmpFer = (((tmpCel*9)/5)+32);

  lcd_1.setCursor(0, 0);
  lcd_1.print(tmpCel);
  lcd_1.setBacklight(1);
}

//Water Sensor
void checkWater(){
  //Check the water level in the bucket.
water = analogRead(waterSensor);
    if (water<=100)
    {
        Serial.println("Water Level: Empty");
        digitalWrite(LATCH_PIN, LOW);
        shiftOut(DATA_PIN, CLOCK_PIN, LSBFIRST, 0b10000000);
        digitalWrite(LATCH_PIN, HIGH);
        digitalWrite(tankPin, HIGH);
      // keep watering for 5 sec
      delay(5000);
      // turn off water
      digitalWrite(tankPin, LOW);
    }
    else if (water>100 && water<=400)
    {
        Serial.println("Water Level: Low");
        digitalWrite(LATCH_PIN, LOW);
        shiftOut(DATA_PIN, CLOCK_PIN, LSBFIRST, 0b11100000);
        digitalWrite(LATCH_PIN, HIGH);
    }
    else if (water>400 && water<=450)
    {
        Serial.println("Water Level: Medium");
        digitalWrite(LATCH_PIN, LOW);
        shiftOut(DATA_PIN, CLOCK_PIN, LSBFIRST, 0b11111000);
        digitalWrite(LATCH_PIN, HIGH);
    }
    else if (water>450){
        Serial.println("Water Level: High");
        digitalWrite(LATCH_PIN, LOW);
        shiftOut(DATA_PIN, CLOCK_PIN, LSBFIRST, 0b11111111);
        digitalWrite(LATCH_PIN, HIGH);
    }
    delay(1000);
}

//moisture sensor
void checkMoisture(){
  //Measure soil humidity
  int moisture = analogRead(soilSensor);
  Serial.println(moisture);
  delay(2000);

  lcd_1.setCursor(0, 1);
  lcd_1.print("Moisture: " + String(moisture) + "   ");
  //Serial.Write(moisture);
  if (moisture <= dry) {
    // the soil is too dry, water!
    watering();
  } else {
    Serial.println("Moisture is adequate. No watering needed " + String(moisture));
    digitalWrite(pumpPin, LOW);
  }
}
//Watering
void watering(){
  int moisture = analogRead(soilSensor);
  Serial.println("Watering now..moisture is " + String(moisture));
    digitalWrite(pumpPin, HIGH);
    // keep watering for 5 sec
    delay(5000);
    // turn off water
    digitalWrite(pumpPin, LOW);
  completeWatering();
  //counter++;
}
//Music
void completeWatering(){
  //Let them know that watering is complete.
    lcd_1.setCursor(0, 0);
      lcd_1.print("   Thank you!   ");
      lcd_1.setCursor(0, 1);
      lcd_1.print("   (ˆ O ˆ)/     ");
      Serial.println("Done watering.");
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,MI,1200) ; // E
      delay(BEAT) ;
      delay(BEAT) ;
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,MI,BEAT) ; // E
      delay(BEAT) ;
      tone(PIN,RE,BEAT) ; // D
      delay(BEAT) ;
      tone(PIN,DO,BEAT) ; // C
      delay(BEAT) ;
      tone(PIN,RE,1200) ; // D
      delay(BEAT) ;
      delay(BEAT) ;
  delay(4400);
  //counter++;
   lcd_1.setCursor(0, 0);
   lcd_1.print("                  ");
   lcd_1.setCursor(0, 1);
   lcd_1.print("                  ");
  ckeckBME280();
  int moisture = analogRead(A0);
   lcd_1.setCursor(0, 1);
   lcd_1.print("Moisture: " + String(moisture) + "   ");
  Serial.println("Soil moisture is " + String(moisture));
}
```

APPENDIX C
REFERENCES

[1] https://create.arduino.cc/projecthub/lc lab/automatic-watering-system-for-my-plants-b73442

[2] https://www.academia.edu/37784635/WATER IRRIGATION SYSTEM USING ARDUINO

[3] https://www.academia.edu/40192969/AUTOMATIC PLANT WATERING SYSTEM USING ARDUINO

[4] https://www.ijert.org/research/automated plant watering system IJERTCONV5IS18012.pdf

[5] https://www.electricaltechnology.org/wp-content/uploads/2018/08/Automatic-Plant-Watering-Irrigation-System-Circuit-Code-PDF.pdf

[6] https://create.arduino.cc/projecthub/neetithakur/automatic-plant-watering-system-using-arduino-uno-8764ba