[1]

Clover

Tucker Chaisit, EE, Omar Yacteen, EE, Brandon Zeng, CSE, and Tim Zhang, EE

***Abstract*—Keeping indoor plants can have many benefits on the human condition such as increased positivity, reduced stress, and even increased focus. Not only are they beneficial to health, but they can also serve to decorate indoor spaces and add beauty to one's surroundings. One of the main barriers to owning an indoor plant is the constant attention necessary to care for it, and plants can die because of too little water and sunlight, or too much. Our design features a compact sensor unit for plants that connect to a wireless hub enabling communication over the internet. Clover gives plant owners the insight into a plant's needs using various sensors and will notify them via an app when and how to care for their plants.**

I. Introduction

The use of plants as interior decoration is a common sight in households all around the world. They are a piece of a living nature which can be introduced to homes. As the daily environments of humans become increasingly urban, more than 85% of a person's daily life is spent indoors [1]. Modern technologies and their rapid developments have affected society in unprecedented ways, and with that disruption includes increased levels of stress as a result. The presence of plants in our living and working spaces will be increasingly important for human wellbeing as technology continues to distance us from nature. Indoor plants have been studied to have various benefits: enhancement of job satisfaction, reduction of psychological stress, improved mood, and enhanced cognitive health [1]. However, plants - being living organisms - require certain means of sustenance: these primarily being light and water. Although these means of sustenance are common amongst almost all plants, different plants grow best under different levels of soil moisture, and lighting conditions. Those who are adept at plant care: "green thumbs" may have acquired their abilities through lots of learned experience, studying, or taking time to monitor the plants. This can be quite a time-consuming task, requires deep knowledge of plants, and can be labor-intensive. Many plant owners simply want the benefits of owning plants without the work to monitor and maintain them.

Existing Solutions

One alternative which is already on the market, are automated growing stations (see Image 1) [5]. These growing stations provide artificial light and can monitor plant conditions including light and water. Although the ability to monitor plant conditions is something which we want to be capable of doing, there are other issues with this design which we intend to change when implementing our plant monitoring system. Firstly, the enclosure itself acts as a monitor and is therefore a fixed size. Additionally, the plant seeds are only available in pre-seeded capsules. Lastly, the automated growing stations have limited programmability and are generally expensive compared to alternatives (~$60-$200).

Image 1: Automated growing station [5].

Another existing solution readily available is the Smart Pot (see Image 2) [6]. Once again, this device allows a user to monitor the watering levels of their plant to ensure that it is ideal. Additionally, it notifies the user of their plant's status via a smartphone app, which is something which we want to implement. However, there are some issues associated with this product. For one, as with the automated growing station, the Smart Pot is a fixed size, and will therefore only support a limited number of plants. Additionally, the Smart Pot lacks programmability for different lighting and watering conditions; something which we would want in a product.

Image 2: Smart Pot [6].

The last alternative design that we found which is already on the market is the Vistefly Flower Smart Care Sensor [7]. This device is the design alternative that offers some desirable

[1]

T. Chaisit from Malden, MA (e-mail: jsathapornch@umass.edu).
O. Yacteen from Amherst, MA (e-mail: oyacteen@umass.edu).
B. Zeng from Amherst, MA (e-mail: bzeng@umass.edu)
T. Zhang from Amherst, MA (e-mail: yumouzhang@umass.edu).

features such as a small form factor, real time analysis, and connect and setup through a mobile application. However, one of the major cons found from the sensor is that it uses a resistive type soil sensor which is known to build up corrosion and decrease the accuracy overtime. It also does not work well in loose soil. This device also lacks programmability and has no notification system.

Image 3: Vistefly Flower Smart Care Sensor [7].

Clover is a product that intends to aid plant owners in caring for their houseplants by periodically monitoring light conditions and soil moisture levels, both of which can be configured for different types of plants by the user. Our design aims to make it easy to take care of plants without having to constantly monitor them. By utilizing a plant-mounted sensor device connected to a wireless hub, the user can maintain their plant worry free.

As with all products which aid in certain skills (in this case, plant care), there are two main impacts a product such as Clover may have on society. On one hand, being able to monitor plant health remotely means that a user, even without prior knowledge of how to care for a certain plant, will have the ability to know if their plant is being placed in ideal conditions to grow. This could extend beyond household plants as well, with plants used to decorate businesses being monitored by groundskeepers if they have access to Clover's notification data.

On the other hand, however, one could argue that a device such as this may act as a crutch to those getting into plant care. That is; some individuals may find themselves reliant on it to monitor plant health, rather than having an intuitive knowledge of how much water or light a plant would require. Similar to how many have become increasingly reliant on spell checkers to write without errors, society may find themselves becoming increasingly reliant on Clover to take care of their plants. We hope that this product will enable the teaching of proper plant maintenance to users so ultimately, they may understand how to properly care for their plants naturally.

Since Clover is a device that is intended to be low maintenance, the user only needs to place the device into a pot of soil, and then program it to whatever plant they would like to monitor. Due to this, the requirements and specifications largely revolve around the device's ability to run without intervention. That is, the device needs to be small enough to fit inside most household plant pots/enclosures, draw small amounts of power such that the device can run indefinitely off of a rechargeable battery being charged by a solar panel, contain a programmable plant library, a notification system for the user, and be water and dust resistant. These specifications are detailed in Table 1.

In order to meet the several requirements listed in Table 1 below, experimentation with various aspects of our design had to be explored. In order to make the enclosure for Clover as compact as possible, our PCB design went through two prototyping phases; the first PCB design focusing primarily on confirming functionality, and the second design based on a tweaked version of the first, but made to be less than half the size (in terms of surface area, see Figure A, below). In order to meet the power requirement, two different methods of powering Clover were explored. One system utilized a NiMH (nickel metal hydride) battery, and the other utilized a LiPO (lithium polymer) battery. Although the method used to charge the NiMH battery is simpler than that of the NiMH battery, the LiPO proved to outperform the NiMH battery so much, that it greatly exceeded our power requirements, and it was therefore decided to use a LiPO battery to power Clover, rather than an NiMH battery in order to meet our power specifications.

Requirement	*Specification*	*Value*
Portable	Compact form factor fits into most plant enclosures	<1 lb, <3 x 2 x 7 in, solar charging
Low Power	Run indefinitely off rechargeable battery, recharged by solar cell	<50 mW
Notifies User	Product will notify users when the plant requires attention	Effective Bluetooth transmission range: 40m with 90% success rate
Dust & Water Resistant	Product needs to be resistant to water and dust contamination	IP64 dust and water resistance rating
Programmable Plant Library	Tracks and stores records of active plants-=0	Supports 10 different plant classes (flowering, palm, foliage, etc.)

Table 1: Requirements and Specifications

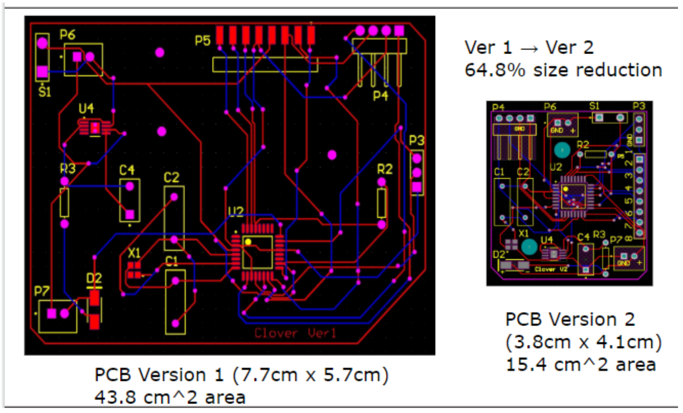


Figure A: First PCB design (left), versus the second PCB design (right).

II. DESIGN

A. Overview

The way we solve this problem is by designing a system that will help the user monitor their plants through their mobile app. The system is divided up into four subsystems: 1) Sensor, 2) Wireless Hub, 3) Cloud Database 4) User Interface. Each subsystem has their own unique functionality which helps us reach our goal.

The main component of our system is the sensor which will handle the data collection through its two sensors operated by a microcontroller that is central to the subsystem. We chose the Arduino Micro platform for prototyping because of its small form factor [8]. The enclosure is powered by a small li-on battery using a step-up converter that also utilizes the Arduino's low power library that helps reduce the power draw of the system significantly by putting the Arduino to sleep. A unique feature of the sensor module is that the power supply is capable of self-charging through its solar cell [9]. It can handle reading light measurements through its voltage level and charge the enclosure at the same time.

To handle communication with the other subsystems, we use the DSD-Tech HM 10 BLE Module (Bluetooth Low Energy) [10] because of its small form factor, wide data transfer range of up to 50 meters, and its low power capabilities. The module can go into sleep mode like the Arduino Micro, which reduces the need to consume energy. In active state, the BLE module uses 8.5mA of current and in sleep it uses up to 200 uA of current, which can help meet the specifications of our design.

The Raspberry Pi [11] acts as a gateway between the enclosure and the MySQL database which stores all the data. We use the Raspberry Pi 4, the latest Pi model available, because of its faster processing speed and similar price range to that of its predecessor, the Pi 3. The Pi secures a connection with the BLE module utilizing the Bluepy library, and it does all the data processing and modifies it before sending a query to insert the data into the MySQL database.

The MySQL database is hosted by Amazon Web Services [12] which is a cloud hosting service that removes the necessity of needing any physical hardware to store data. AWS is also used to host our web server which helps handle our data requests from the user interface to the MySQL database.

The user interface will be run on an Android app called Clover, which contains multiple features. Here, users will be able to monitor real-time data being recorded from their sensor module regarding their plant's condition like moisture level and light. A main feature is that users can set notifications to keep them reminded of when their plant requires sustenance. There is also a database that contains pre-chosen plants that can be used to keep track of ideal plant conditions. If a user has their own personal plant, they have the capabilities of adding their own custom plant with their own certain ideal parameters for its growth to the database. Also, users can see a visualized graph of past measurements read and stored in the database.

The reason why we expect that this technology will solve the problem is because with the completed system, the technology will be able to report a live condition of the plant and send a notification report to the user through the application whether or not the plant needs more care from the user, and that way the user will feel more at ease about taking care of their plant. An overview of this system is outlined in the block diagram shown in Figure 1 below.

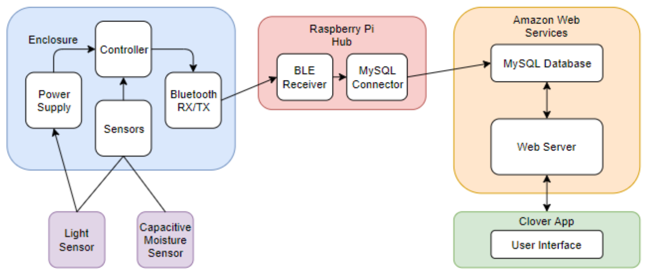


Figure 1: Block Diagram of Clover.

B. Block 1 – Enclosure/Sensor Peripherals

The enclosure contains the microcontroller, power supply, and Bluetooth module. The peripheral sensors, which communicate to the controller, are what allow Clover to extract meaningful information from its environment. For the MDR demo, two sensors were used: a capacitive moisture sensor - which was used to measure soil moisture levels - and a solar panel, from which voltage output is used to measure light intensity levels (see Figure 2b). As seen by Figure 2b below, for various levels of light intensity received by the solar cell, a corresponding voltage value is generated. These voltage values can then be calibrated to the corresponding light intensity values, and used by Clover in order to determine light intensity levels.

In addition to inferring light intensity, the solar panel is also used to recharge the onboard battery cell, which can recharge the battery cell completely within 15 hours even in less than ideal lighting conditions (see Figure 2a). The battery cell is considered fully charged at 1.25V, which occurred at ~15 hours of charging at 3,400 lux (lower end of medium light intensity). A charging circuit based around a LiPO integrated circuit was used in order to ensure that the onboard battery battery is not charged at unsafe current levels (Figure 2c). For the purposes of charging Clover's onboard battery, the charging voltage was limited to a trickle charge of no more than 3 volts.

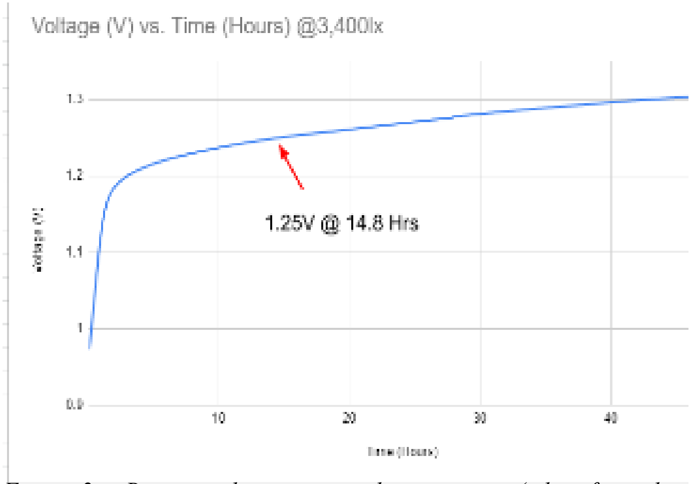


Figure 2a: Battery voltage versus charging time (taken from the initial design. using NiMH batteries)..

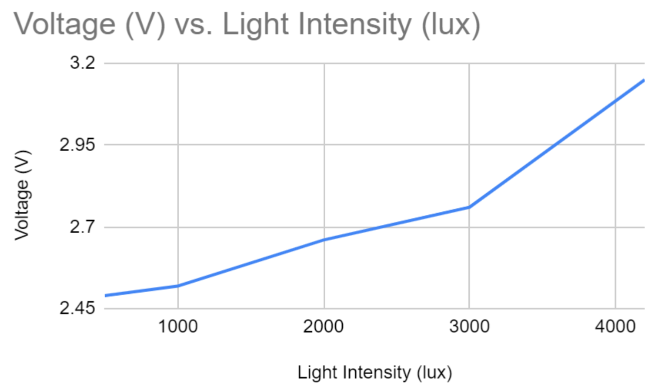


Figure 2b: Mapping of voltage vs light intensity in lux.

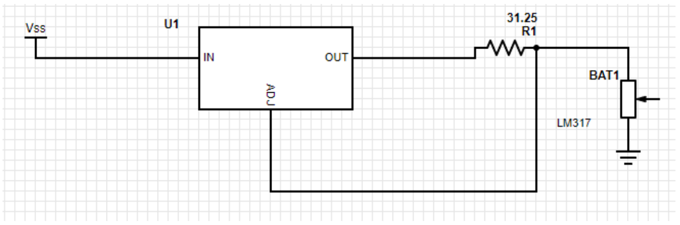


Figure 2c: Trickle charging circuit.

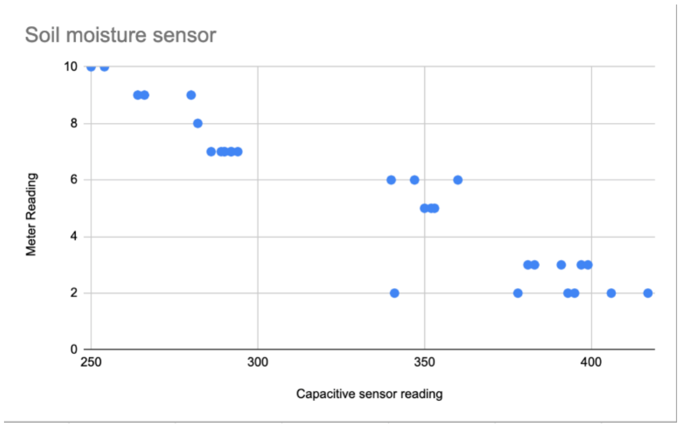


Figure 3: Mapping of meter reading vs capacitive sensor reading.

In terms of calibrating the soil moisture sensor, we decided to take an approach of doing an experiment to match the value from the capacitive sensor to a moisture meter, which is a commonly used tool in the household to measure the soil moisture level. By taking this approach we can define the meaning of the value from the capacitive sensor as well as categorize those values for each plant in the database. A graph of meter readings against capacitive soil values are mapped in Figure 3. In Figure 3. Y axis represent a scale from the meter and the X axis is the value from our sensor. Note that the higher the meter reading value, the more moisture content is present in the environment. This means a lower capacitive sensor value correlates to a wetter environment. The graph show a distribution of correlated value, and that way we are able to define the value that read from our sensors into something that's meaningful

C. *Block 2 – Raspberry Pi Hub*

The Raspberry Pi hub is implemented to act as a gateway between the enclosure and the MySQL cloud server. The hub contains a python script that will run on system boot, where it will connect to the MySQL database, search for the BLE module's Bluetooth address, and automatically connect to the device. After connection, the hub will then stay in "listening" mode awaiting any new data stored in the BLE buffer. If new data is read, the hub will process the data and send a query to insert new data into the MySQL database.

D. *Block 3 – Amazon Web Services*

This subsystem represents all the services being used from AWS. AWS offers quick and efficient cloud storage services. In our project, we created a MySQL database which contains two tables, plant monitor and plant database. The plant monitor table contains all the read sensor measurements with the date and time it was read. The plant database includes all the plant's and their ideal growth conditions. Currently this database is designed to handle one user, so if one user adds a plant to the database, others who have the app can see that plant added as well. We plan on implementing ways to have the database to accommodate for different users by including user ID tags. We created a cloud web server that is used to handle requests from the user interface. The web server is run using ubuntu, and there we installed Apache and PHP packages to help us store and run PHP scripts. The scripts' functionality is to connect to the MySQL database and manipulate the data by handling the user's requests and return the data in the form of a JSON array to the user interface.

E. *Block 4 – User Interface*

The user interface is where the user will be interacting with Clover. The app contains features where it can allow users to monitor their plants. The app contains an active plant library that is stored in the database, where users have the option to choose a pre-existing plant in the library or add their own custom plants to control the conditions of how they want their plants to grow. It also provides a history graph of the active's plant nutrient level from the previous days before. The app receives and sends data to/from the MySQL database through the PHP scripts stored on the web server.

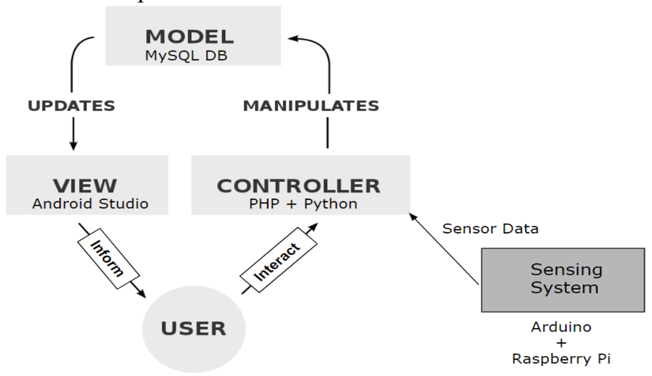


Figure 4: Software Flow Diagram.

Expanding on the role of each block. Looking at Figure 4, this is a representation of a MVC diagram (Model - View - Controller). making some modifications to the original diagram for our project. This diagram provides an overview of how data is being passed/stored along the Clover system. The "Model" being our MySQL database is where our data is being stored/managed. When the new data is updated to the database, it updates the "View" or Android app to inform the user of the change. If the user wishes to modify the plant library, they interact with the "Controller" which stores the PHP and Python scripts to make GET/POST requests to update the database. On the right side, the sensing system is its own block that interacts with the "Controller" adding new data into the database.

III. THE PRODUCT

A. Product Overview

Referring back to the block diagram of Clover, we see that the system begins with measurements taken from the soil moisture sensor, and from the solar panel (see capacitive moisture sensor and top-mounted solar panel in figure C, below). The data from these sensors is fed through the microcontroller, inside the enclosure (see 'PCB -1' in figure C, below). For the solar panel, power is also provided to a charging circuit, in order to ensure that the battery remains charged. The data from the sensors is then transmitted via a Bluetooth transmitter (see 'PCB-1', in figure C, below), to be received by a Raspberry PI. Once received by the raspberry PI, the data is compared against information on a remote MySQL database. Based upon user-specified settings, Clover would then periodically notify the user over the internet if their plant was within recommended values for lighting conditions, as well as soil moisture conditions. The user would receive these notifications, and specify settings through a dedicated app (see figure 4, above).

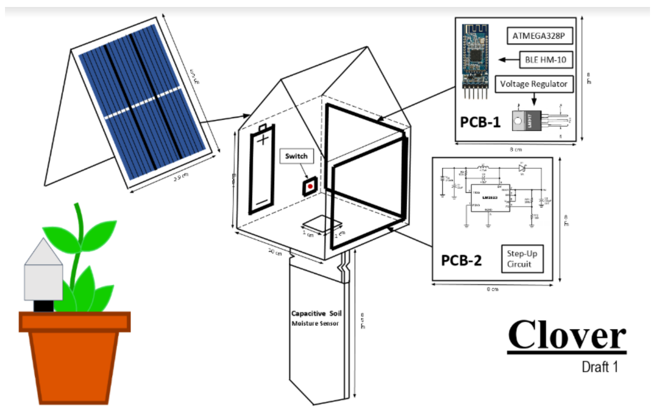


Figure C: Clover Product sketch (version 1). Note: This sketch was created while still discussing the possibility of using two PCBs, which was scrapped in favor of using a single PCB.

B. Electronic Hardware Component

Although the use of two separate PCBs was discussed (with one PCB housing the main hardware for Clover, such as the microcontroller, Bluetooth transmitter, etc. and the other PCB housing power related circuitry), it was ultimately decided to simply integrate all of the electronics of Clover onto a single PCB. As mentioned in section 1, above, there were two PCB designs that were created to arrive at our final design (see figure A, in section 1). The first PCB was created primarily to test functionality. The second PCB was then created based off a modified version of the first PCB and was made to be much more compact (see Figure A, in section 1). The first fully assembled and tweaked PCB design is illustrated below, in Figure B. Unfortunately, due to unforeseen time constraints, our team was unable to assemble our second PCB design.

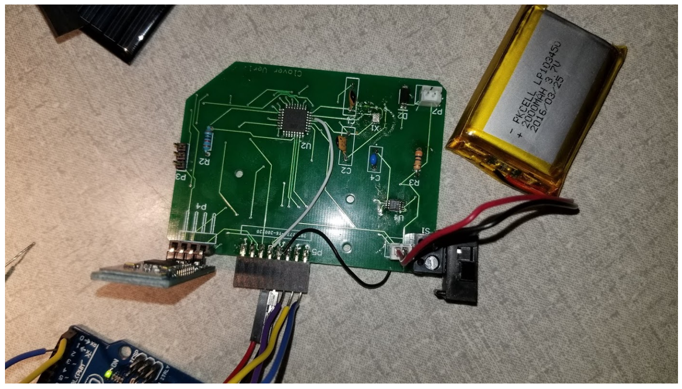


Figure B: First fully-assembled PCB design.

C. Product Functionality

Although the final version of our app was not yet created, our programmable library still required some development, and our second, smaller PCB design was not yet assembled or tested, Clover's core features were still functional: Clover was able to receive information regarding the plant's soil moisture levels and lighting conditions, and transmit that data to be processed remotely, as well as to a smartphone app. If given additional time, efforts to finalize the second PCB design would have been realized and work would have shifted focus on building out the software components of the system.

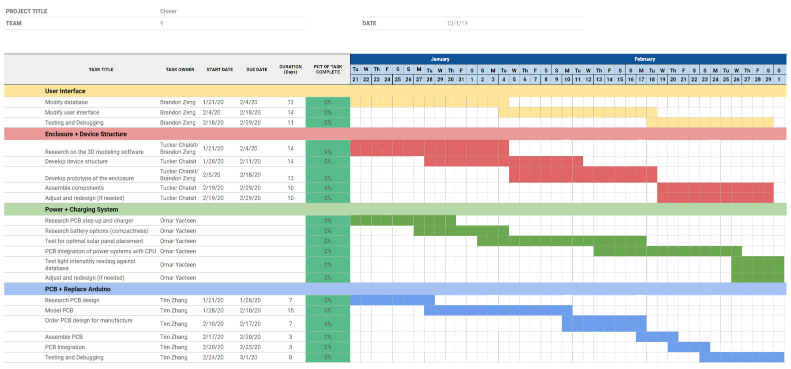
D. Product Performance

Although most of our requirements from Table 1 were met, there were some requirements which we were unable to test or fulfill. For example, although we were able to test the power requirements, enclosure size, plant library, and app notifications, and could confirm that Clover met those specifications, we were unable to test those requirements relating to Bluetooth transmission range, or the waterproofing requirements. These requirements could be tested relatively easily, however, and our team was able to meet most of our product requirements and specifications.

Table 2: Project management Table

System	***Controller and Hub Integration***	***Power & Solar Cell Integration***	***Capacitive Moisture Sensor***	***Database and App***
Team Member	Tim	Omar	Tucker	Brandon
MDR Deliverables Completed	1. Processing of data from connected sensors 2. Transfer processed data to database via hub	1. Solar panel can successfully, and safely, charge battery. 2. Battery can be used to power arduino. 3. Solar panel can be used by arduino to infer lighting conditions.	1. The capacitive moisture sensor is able to collect data	1. Create functional MySQL database and webserver. 2. Implement Restful API to have user interface communicate to database. 3. Create User Interface.
What's Left	1. Integration of sensor device components onto custom PCB	1. Reduce power consumption of device to allow Clover to run continuously. 2. Research alternative, low power light sensor options (for more accurate readings). 3. Integrate charging, step-up, and solar cell circuits into main PCB.	1.Perform more experiment trial to make the reading more accurate. 2. Research way to help cut down the power drawn by the sensors.	1. Improve functionality of app to handle different users 2. Research on making an enclosure

CDR GANTT CHART



PROJECT TITLE	Clover
TEAM	9
DATE	12/1/19

TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION (Days)	PCT OF TASK COMPLETE
User Interface					
Modify database	Brandon Zeng	1/21/20	2/4/20	13	0%
Modify user interface	Brandon Zeng	2/4/20	2/18/20	14	0%
Testing and Debugging	Brandon Zeng	2/18/20	2/29/20	11	0%
Enclosure + Device Structure					
Research on the 3D modeling software	Tucker Chaisit/ Brandon Zeng	1/21/20	2/4/20	14	0%
Develop device structure	Tucker Chaisit	1/28/20	2/11/20	14	0%
Develop prototype of the enclosure	Tucker Chaisit/ Brandon Zeng	2/5/20	2/18/20	13	0%
Assemble components	Tucker Chaisit	2/19/20	2/29/20	10	0%
Adjust and redesign (if needed)	Tucker Chaisit	2/19/20	2/29/20	10	0%
Power + Charging System					
Research PCB step-up and charger	Omar Yacteen				0%
Research battery options (compactness)	Omar Yacteen				0%
Test for optimal solar panel placement	Omar Yacteen				0%
PCB integration of power systems with CPU	Omar Yacteen				0%
Test light intensitity reading against database	Omar Yacteen				0%
Adjust and redesign (if needed)	Omar Yacteen				0%
PCB + Replace Arduino					
Research PCB design	Tim Zhang	1/21/20	1/28/20	7	0%
Model PCB	Tim Zhang	1/28/20	2/10/20	15	0%
Order PCB design for manufacture	Tim Zhang	2/10/20	2/17/20	7	0%
Assemble PCB	Tim Zhang	2/17/20	2/20/20	3	0%
PCB Integration	Tim Zhang	2/20/20	2/23/20	3	0%
Testing and Debugging	Tim Zhang	2/24/20	3/1/20	8	0%

IV. CONCLUSION

As of now the core data path from data collection via sensors to the displaying of this data in the app is functional. We can employ the soil moisture sensor and the solar cell to collect moisture and light data periodically from the microprocessor unit. This data is then sent wirelessly from the microprocessor via Bluetooth to our Raspberry Pi that can preprocess the data before uploading it to the cloud database using Wi-Fi. From there, the data can be queried by the app and displayed on the user interface. With the underlying framework operational, we have a greater understanding of the next steps needed to realize the final product.

Going forward, the largest challenges ahead will be integrating the prototype into its proper form factor and optimizing the energy efficiency of the sensor unit. It will be necessary to integrate all the functionality of the Arduino microprocessor onto a custom PCB including sensor control, power management, Bluetooth communication, and sleep capability. This will heavily involve the use of PCB modeling software such as Altium Designer, where we need to be sure that all the required components are integrated before sending it off for manufacture. We also will have to design the enclosure that can both meet the size and durability specifications, which will involve the use of a 3D printer and software such as SolidWorks. Finally, the energy consumption of the sensor unit should be low enough that the battery on board maintains charge just from the energy produced by the solar cell when regularly exposed to sunlight. Possible approaches to solving this problem include design considerations in the PCB model, modifying the size of the solar cell, modifying the battery capacity, and modifications to both hardware and software components of the final product.

ACKNOWLEDGMENT

Team 9 would like to thank our advisor, Professor

Christopher V. Hollot for his thoughtful guidance and practical advice throughout the design process. We would also like to thank our evaluators Professor Kundu and Professor Xu for their valuable feedback and willingness to help guide our team.

REFERENCES

[1] M. Lee, J. Lee, B.-J. Park, and Y. Miyazaki, "Interaction with indoor plants may reduce psychological and physiological stress by suppressing autonomic nervous system activity in young adults: a randomized crossover study," Journal of Physiological Anthropology, vol. 34, no. 1, p. 21, Apr. 2015.

[2] Gupta, Devashish. "Capacitive v/s Resistive Soil Moisture Sensor." Hackster.io, 12 July 2018, www.hackster.io/devashish-gupta/capacitive-v-s-resistive-soil-moisture-sensor-e241f2.

[3] "Tutorial - Using Capacitive Soil Moisture Sensors ." *Tutorial - Using Capacitive Soil Moisture Sensors on the Raspberry Pi* -, 10 Nov. 2018, www.switchdoc.com/2018/11/tutorial-capacitive-moisture-sensor-grove/.

[4] "How to connect to MySQL using PHP." a2hosting, n.d. www.a2hosting.com/kb/developer-corner/mysql/connect-to-mysql-using-php

[5] Click & Grow Smart Indoor Garden, www.amazon.com/Click-Grow-Smart-Garden-Indoor/dp/B01MRVMKQH/ref=asc_df_B01MRVMKQH/?tag=hyprod-20&linkCode=df0&hvadid=229286241014&hvpos=1o4&hvnetw=g&hvrand=6217245732423849610&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=1018329&hvtargid=pla-393445428625&psc=1

[6] Parrot Pot Smart Flower Pot, www.amazon.com/dp/B01KV0JCNY/ref=twister_B01MEF3FEN?_encoding=UTF8&psc=1

[7] Vistefly Flowers Smart Care Sensor, www.amazon.com/Vistefly-Moisture-Sunlight-Fertility-Gardening/dp/B07NRYK2DB

[8] Atmel, "8-bit Microcontroller with 16/32K bytes of ISP Flash and USB Controller" ATmega16U4/ATmega32U4 datasheet, 2015.

[9] AMX3d Micro Mini Solar Cell, https://www.amazon.com/AMX3d-Micro-Power-Panels-53X30mm/dp/B01BFX65YK/ref=sr_1_4?keywords=amx3d+micro+mini+solar+cell+5v+30ma&qid=1572558496&sr=8-4

[10] DSD Tech, "HM Bluetooth Modules" HM-10 datasheet, January 2017.
[11] Raspberry Pi Ltd. "Raspberry Pi 4 Model B" datasheet, June 2019.

[12] Amazon Web Services, aws.amazon.com

[13] "IP Rating Chart," DSMT.com. [Online]. Available: http://www.dsmt.com/resources/ip-rating-chart/. [Accessed: 11-May-2020].

E. *Design Alternatives*

A variety of design alternatives were proposed when first conceptualizing how exactly we were going to meet our product specifications. Initially, we had tried to keep the solution as simple as possible with just the development of a single sensor device capable of collecting the data and transmitting it via only Wi-Fi. We had chosen the ESP32 Wi-Fi module as a commonly used low power wireless adapter for Arduino boards to enable wireless communication to our server and plant database. When it came to present this first idea, it was met with criticism in that Wi-Fi's power consumption would be too high for our low power sensing purposes.

The next proposed idea would be to simply replace the Wi-Fi module with a Bluetooth adapter to enable wireless communication. This allowed for lower power consumption when transmitting data, but the drawbacks were that it has a lack of internet connectivity. Therefore, much like the Vistefly product, the device would not be able to notify users without directly pairing to the sensor.

To satisfy our design specifications and have all the features we desired, we ultimately decided on the use of a Bluetooth enabled sensor device paired to a wireless hub capable of internet connectivity. This became our final design plan that we are currently working on.

F. *Technical Standards*

IEEE 802.15.1-2002 - IEEE Standard for Telecommunications and Information Exchange Between Systems This IEEE Standards product is part of the 802 family on LAN/MAN. Abstract: The lower transport layers [(Logical Link Control and Adaptation Protocol (L2CAP), Link Manager Protocol (LMP), baseband, and radio] of the Bluetooth™ wireless technology are defined. Bluetooth is an industry specification for short-range radio frequency (RF)-based connectivity for portable personal devices.

IP64 Water and Dust resistance - Technical standard used to denote a device which is completely resistant to dust ingress, as well as water spray in any direction [13]. Of course, given that Clover would be exposed to water spray and dust, meeting this technical spec would be ideal. Unfortunately, due to unforeseen time constraints, our team was unable to complete the testing required to confirm this level of water and dust resistance.

G. *Testing Methods*

One experiment that we executed was testing the battery charging circuit with LabView. In order to ensure the battery was being charged safely by the solar cell, a trickle charging circuit was designed, which limited the current being used to charge the battery. However, in order to test this circuit, a depleted battery needed to be recharged over a long period of time, and its voltage periodically measured to track its depletion level. In order to test and track the charging circuit, a script in Labview was created, which connected the SDP lab computer to a Keysight multimeter in the SDP lab. This script would automatically record the voltage measured at the battery and store the data in a text file. The experiment itself then consisted of charging a depleted battery for about 50 hours over a weekend, with the Labview script recording measurements once every 10 minutes, and then plotting the recorded values in a spreadsheet (see Figure 1 above). We were then able to use this plot to track the performance of our loaded trickle charger.

In order to confirm that the system meets power specifications, a similar experiment to the trickle charger test will be done; whereby the current draw will be monitored periodically (two measurements can be taken to get a good average: one during sleep mode, and one during wake mode), and plotted. When multiplied by the voltage our circuit is running at, this will allow us to track the power consumption of Clover, allowing us to numerically confirm with we are meeting the average power limit of 50 mW.

The experiments that we did were testing and match the value of the moisture meter and our capacitive soil sensor to make the value we read from the capacitive soil sensor meaningful and customizable for each plant. The result of the experiment is shown in Figure 3. From Figure 3, we can define the range of the capacitive value into the range of 1 to 10. The value of 1-3 is dry, 4-7 is moist and 7-10 is wet. Further testing is recommended to help make the reading value more accurate.

H. Team Organization

Brandon is the software lead since he is the single CSE of the group and has had the most exposure to software engineering. Omar is the power circuit lead since he is responsible for implementing the solar cell and has had the most experience working with power supplies. Tucker oversees one of the core sensors in the project but also assists other members with whatever tasks are needed. Tim is responsible for integration of the controller hardware with the software given his experience in robotics and interest in software.

Overall, the team has been functioning well except for a variety of scheduling conflicts, with every assignment and deadline met. Team cohesion needs to be improved given it has been difficult this fall semester to find long periods of time where our group can work together on the project. Many of us work while also attending school and our work hours often present time conflicts. We will need to discuss ways of optimizing our time when we meet and possibly plan at least two periods per week where the whole team can work deeply together. Aside from this our working prototype of the system was shown at MDR with favorable feedback from the evaluators.

I. Beyond the Classroom

Omar: Having interned as an automation engineer last summer, I realized how valuable automating certain processes can be. During my testing of the trickle charger sub circuit for this project, which regulated current for charging the battery cell safely, I realized that generating a plot of battery voltage as the solar panel charged the battery, versus time, would be very time-consuming; taking almost 15 hours to fully charge the battery (see Figure 1). In order to automate this testing process, I downloaded a free trial of Labview, downloaded drivers which allowed the Keysight multimeters used in SDP lab to Labview, and wrote a script which allowed for automated voltage measurements of the battery while being charged. In addition to perfectly spaced measurements, this saved me from needing to "babysit" the testing process, and further demonstrated to me how valuable automation is.

In addition to Labview proving to be a valuable resource, so was Texas Instruments. I designed both the trickle charging circuit, as well as the five-volt step-up circuit, around Texas Instruments integrated circuits (Using the LM317 for the trickle charger, and the LM2627 for the five-volt step-up circuit). I encountered several issues when designing an efficient step-up circuit for this project, and so I resorted to contacting Texas Instruments on more than one occasion. The engineers who I was put in touch with were both quick to reply and were a great help in trouble shooting my circuits.

Tucker: Having experience with sensors from past projects helps with the process of choosing and integrating the sensors into systems. I learned about various kinds of moisture sensors, and the pros and cons of each sensor in order to choose the sensors that would fit and perform what was needed for our systems. The resources that have been useful to me are mostly the online documentation of the project of people that have experience with various types of soil sensors and their way of integrating, testing, and using the sensors in their system. In my opinion, I do not see any connection between this and my life as a professional yet, but definitely for the next semester where we start to implement PCB, 3D printing, enclosure, I believe that I will be able to relate more into life after graduation from there.

Tim: As my primary responsibilities include the programming and integration of the sensor module and wireless hub, I have had to use both a combination of software and hardware skills to successfully complete my tasks. Even though I have had previous experience programming with Arduino, I have had to learn a great deal of information regarding wireless communication through Bluetooth Low Energy. Certain things such as communication protocols, interpreting data packets, and having two separate devices locate each other took time to learn and implement. It was interesting to learn, and I enjoyed the time spent learning the Linux command line and Python programming language for practical applications such as this project. I view the experience of learning things from scratch and quickly implementing them for a project as a critical skill that will be useful going forward as an engineer of any kind.

Brandon: Having prior experience in programming in Java from Freshman and Sophomore year, I was able to use some of those skills and apply it in Android app development. I have had no prior knowledge of making an app, so starting off was quite a task. Slowly learning through online resources like Stack Overflow and YouTube helped ease the process of learning about different android app properties and subcomponents. I also learned a lot about networking with setting up a web server and general MySQL commands. I view these experience quite useful in my future career as many software engineers, require these types of skills in the future.