November 7, 2015

Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

RE: ENSC 440 Design Specification– E-Garden

Dear Dr. Rawicz:

In regards to the course requirement of ENSC 305W/440W, enclosed to the letter is Smart Garden Incorporation's design specification for E-Garden. We are designing and implementing an auto-watering system that enables people to get real time information of their lovely plants and maintain their plants through Internet.

Our design specifications refer to a previous document "Functional Specification for E-Garden", and discuss the design details to achieve the functional requirements [1]. The purpose of this Design specification is to present a detailed description of all technical aspect of the product. This document will also specify, in detail, both hardware and software aspects of the design, including details about accuracy of each component, and the control system design procedure. A practical test plan will also be provided for further clarification.

Smart Garden Incorporation is a well-balanced team by six SFU senior engineering students: Timmy Kwok, WeidiZhai, Duling Lai, Siyan Chen, Bo Sun and Tianguang Zhang, who are very reasonable and motivated students, with various engineering backgrounds. If you have any further questions or concerns about the functional specification, please feel free to contact our CEO Timmy at 778-316-2022 or e-mail sumyuek@sfu.ca.

Yours Sincerely,

Timmy Kwok
CEO
Smart Garden Incorporation

Enclosure: Design Specification for E-Garden System

Design Specification
E-Garden System

Project Team members: Timmy Kwok
Duling Lai
WeidiZhai
Siyan Chen
Bo Sun
Tianguang Zhang

Contact Person: Timmy Kwok
sumyuek@sfu.ca

Submitted to: Dr. Andrew Rawicz
Steve Whitmore
School of Engineering Science
Simon Fraser University

Date: Nov 10, 2015

EXECUTIVE SUMMARY

Our product, E-Garden, is based on the idea of saving time and money on maintaining garden. With this idea in mind, we come up with E-Garden, which enables people to get real time information of their plants and maintain their plants anytime from anywhere. This document details the design specifications for the entire E-Garden system as well as each of the individual components. The goal is to give the reader a detailed look at each portion of the proof-of-concept model from hardware to firmware/software and including specifications and justifications of the design approaches.

The E-Garden System consists of three major parts: Web app, Control System and watering system. The Web app is designed for user to control their backyard garden by using website on any electronic devices. In our control system, we constructed with microcontroller, soil temperature sensor and moisture sensor. Those sensors will send data to microcontroller through Wi-Fi, and analyzed by microcontroller, then sent back to web app. our web app will provide information such as soil temperature, moisture level and watering history. This web app can send data to microcontroller by Wi-Fi in order to adjust the watering system.

The final section of the document provides a set of preliminary test procedures to examine the model functionality of the proof-of-concept model. The test plan is divided into two parts: individual component testing and an evaluation of the integrated system.

The project will be a span of 4-month of challenging work. The prototype design with its specific features will be ready by Dec 1, 2015. We believe that our system with its affordable price range and incredible quality can accommodate many customers in the market.

Table of Contents

GLOSSARY

APP	Short for application; a program which runs in user space
PHP	Personal Home Page； it now stands for PHP: Hypertext Preprocessor, which is a recursive backronym.
HTML	HyperText Markup Language
CEO	Chief Executive Officer
CTO	Chief Technology Officer
MySQL	An open-source relational database management system
URL	Uniform Resource Locator
Wi-Fi	A local area wireless computer networking technology
Bluetooth	A wireless technology standard for exchanging data over short distances
PaaS	Platform as a Service
JSON	Java Script Object Notation
EC2	Elastic Compute Cloud
MSB	Most significant bit
NTC	Negative temperature coefficient
GND	Ground
VCC	Volt Current Condenser
UNO	A type of Arduino PCB
RH	Relative Humidity

1. INTRODUCTION

The E-Garden is aimed to bring the best convenience to maintaining garden. Aside from automatically watering the plant, our product also collects and displays soil humidity and temperature data of your garden. Our web APP, which includes web server and web interface, will enable you to maintain your garden anytime from anywhere. E-Garden is a system embedded with the following main parts (as shown in Figure 1):

- Humidity and Temperature sensors
- Microcontroller
- Servomotor
- Web APP

This document lays out the design requirement for our E-Garden system in order to provide a comprehensive reference for this user friendly product.

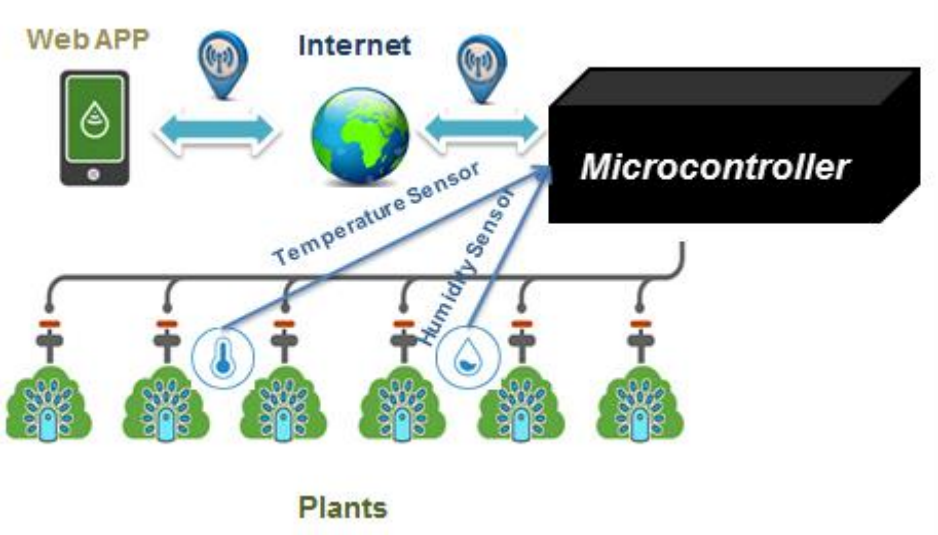


Figure [1]: E-Garden Model

1.1 SCOPE

This document describes the design details of a wireless irrigation system named E-Garden. This design specification includes the design approaches, the final product, the requirements for a proof of concept system, and a set of test plans for each module in the functional specification. These technical details will be used as reference throughout the design phase.

This design specification will outline the following details:

- The E-Garden product and its features

- Overview of the system specification
- Technical details of system design to support the functional requirements
- Test plans to examine the model functionality

1.2 INTENDED AUDIENCE

The intended users of this design specification include all members of our team. The team leaders, CTO and CEO, can use this document as a guide to measure the overall progress. Hardware and software developers of E-Garden will refer to this document to monitor the progress and aim to satisfy the requirements of the system during implementation phase.

1.3 REQUIREMENT CLASSIFICATION

The requirements referenced throughout this document are taken from the functional specification document [1]. The following convention has been used to represent the functional requirement:

[Req x.y.z - pn]

Where 'x', 'y', 'z' indicates the functional requirement number and section; 'p' represents the priority of the functional requirement. The priority is divided into three levels:

- P1 - These requirements are high priority and apply to the proof-of-concept system.
- P2 - These requirements are medium priority and apply to both the proof-of-concept system and the final product.
- P3 – Low priority. These requirements apply to the final product only.

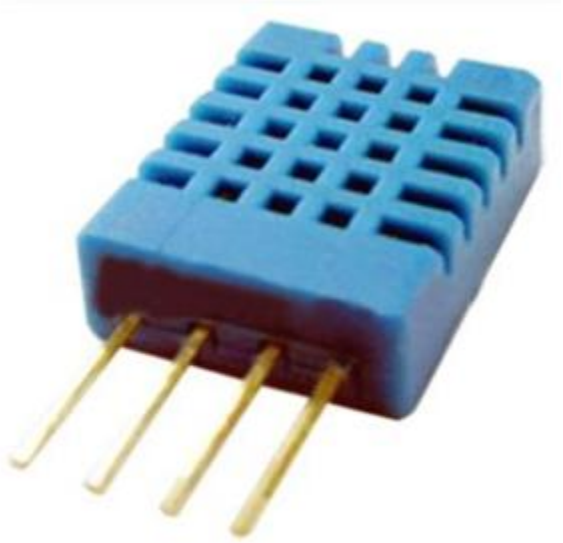
2. System Specification

2.1 HARDWARE DESIGN

2.1.1 SENSOR DESIGN

2.1.1.1 DHT11 TEMPERATURE SENSOR

For temperature detection, the DHT11 temperature sensor, which is compatible with Arduino UNO, is used in the system (DHT11 is shown in figure [2]). The sensor is exposed in the air, and features a temperature sensor complex with a calibrated digital signal output. This sensor includes an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller (Arduino UNO), offering excellent-quality, and fast response temperature detection system.



\

Figure [2]: the DHT11 temperature sensor

In the table [] below, shows the specific characteristics of DHT11 sensor [2]

Supply Voltage	3.5V-5V DC
Supply current (running)	0.5mA typ. (2.5mA max.)
Supply current (stand-by)	100uA typ. (150uA max.)
Temperature range	0 / +50 ℃ ±2 ℃
Interface	Digital
Dimensions	1.05” x 0.7” (connectors excluded)
Weight	0.1 oz (2.7g)

Table [1]: Characteristics of DHT11

Sensor physical interfacing is realized through a 0.1” pitch 3-pin connector: +5V, GND and data. As shown in figure [3], the first pin and the fourth pin are power supply and ground and they are used to power the sensor, the second one is the sensor digital output signal.

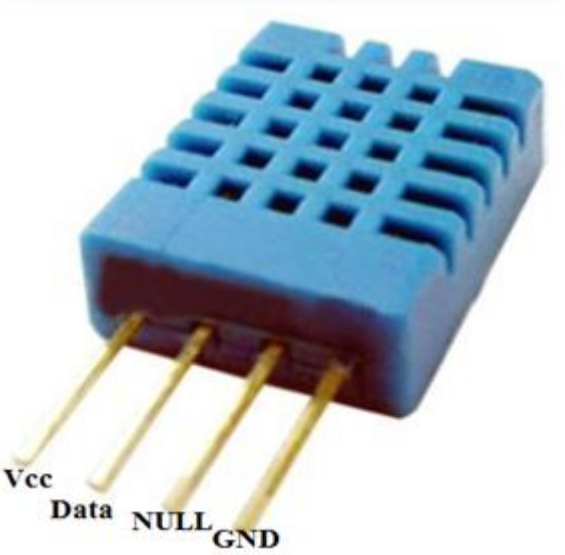


Figure [3]: the pins of DHT11 sensor

To connection with the microcontroller, the single-wire bus needs a 5K ohm pull-up resistor as a voltage divided when the connecting cable is shorter than 20 meters, just as shown in figure [3] below.[4]

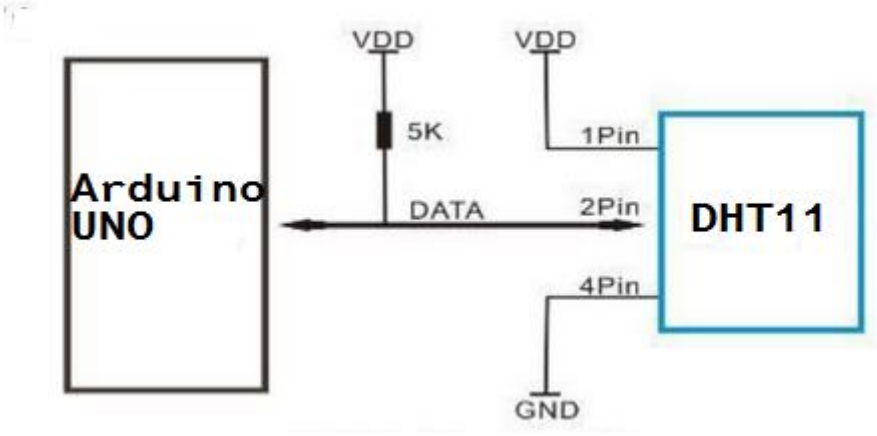


Figure [4]: Connection between DHT11 and Arduino UNO

After DHT11 is powered up, it goes in low power standby mode and it waits to recognize a Start Signal on the DATA line. Once DHT11 detects the Start Signal, it will send out a low voltage level response signal, which lasts 80us. Then it sets the voltage level from low to high and keeps it for 80us.Now data transmission will start. Every bit of data begins with the 50us low voltage level and then switches to high voltage level; high voltage level duration depends on the bit value that has to be transmitted: a 1 bit has a high voltage level duration of 27us, a 0 bit has a high voltage level duration of 70uS (as shown in figure [4]).[5]

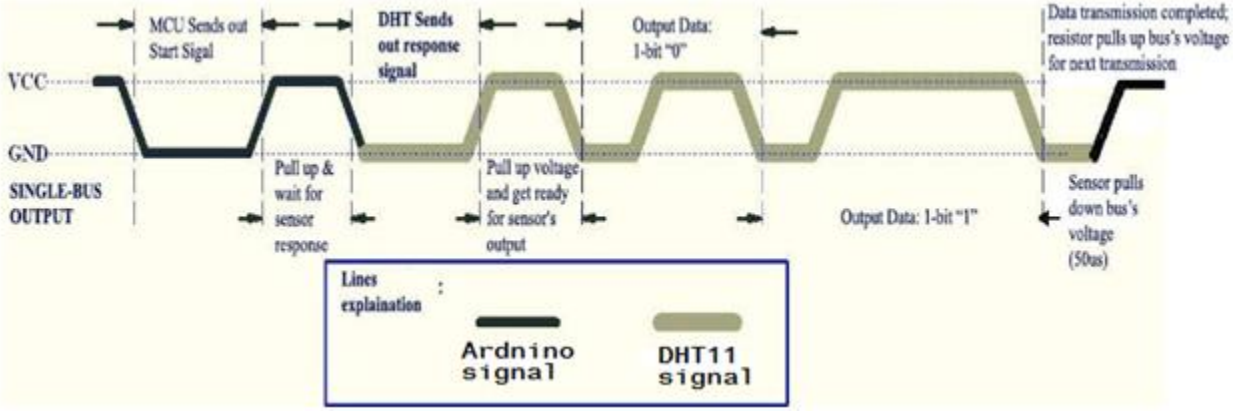


Figure [5]: Transmission between Arduino and DHT11

A complete data transmission is 40bit, so a communication process is about 4mS. DHT11 sensor sends higher data bit first (MSB) in the following format:[6]

Data = 8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data+ 8bit check-sum

If the data transmission is right, the check-sum will be the last 8 bits of the sum of the four byte transmitted.[7]

2.1.1.2 YL-69 SOIL MOISTURE SENSOR

For the soil moisture sensor part, the YL-69 moisture sensor is used. As shown in figure [6], the moisture sensor has two probes; and these probes are used to measure the soil moisture in the soil by telling how well the electrical current is passed between them [7]. The resistance of soil is proportional to the moisture in the soil. When the water in the soil is in shortage, the electrical resistance of the soil will be higher, and this gives a higher output voltage.

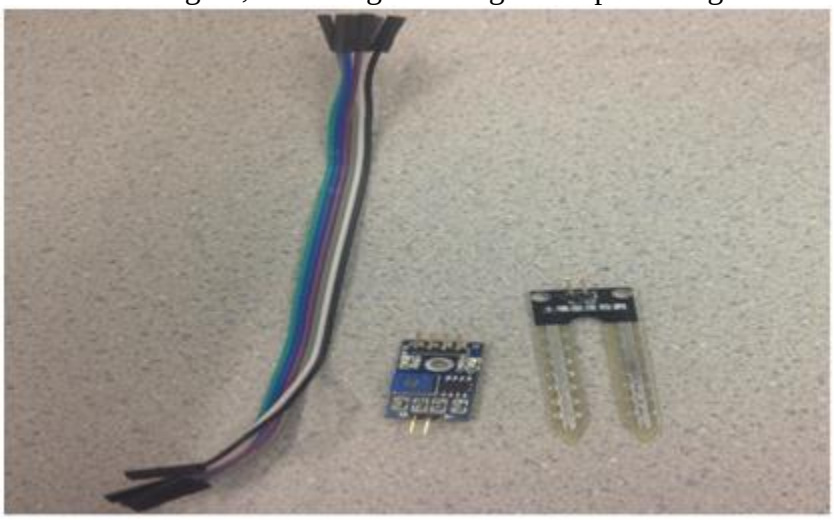


Figure [6]: YL-69 sensor and the Potentiometer (blue)

The moisture sensor with LM393 comparator chip, model YL-69 is used and it comes with a digital potentiometer (blue) as shown in figure [6]. The sensor consisted of 4 pins: an analog output pin 1, a digital output pin 2, a Vcc input pin 4 and a ground pin 3 (as shown in figure [7] below). The sensor can be powered up by connecting the Vcc input pin to the 5V pin in the microcontroller.[R8]

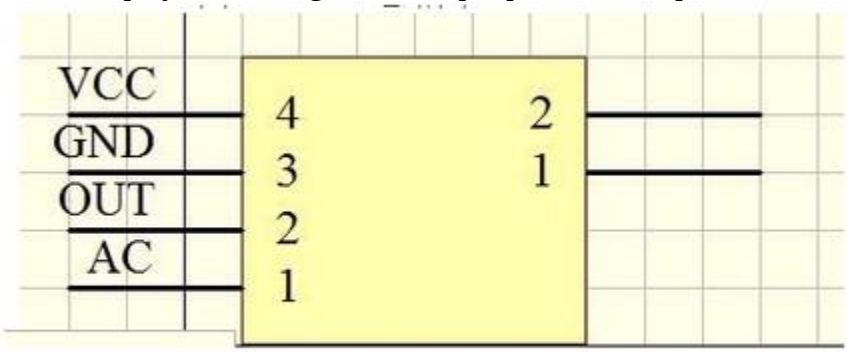


Figure [7]: the pins of LM393

The figure [8] below shows the schematic for the moisture detection sensor module [9]. Analog output is the voltage measurement between two probes of the sensor. For the digital output, it gets set as HIGH when the measured value is greater than the threshold set by the potentiometer and it gets LOW when the measured value is lower than the threshold. However, due to the fact that the digital output is not accurate, the analog output will be used as the analog input to the microcontroller directly connected through jumper wires[10].

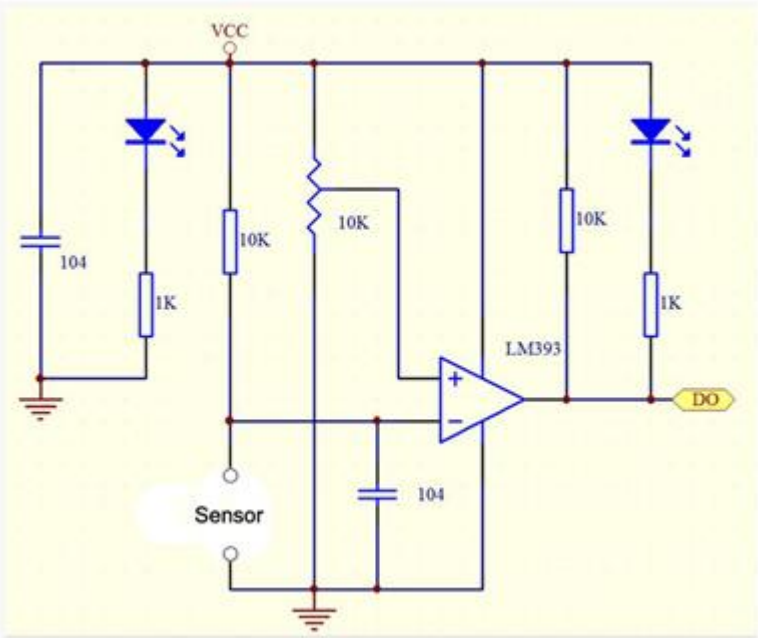


Figure [8]: Schematic Diagram for the Moisture Detection Sensor Module[11]

The connection between the sensor and the microcontroller is shown in figure [9] below. The Arduino UNO microcontroller has an onboard 10-bit 6-channel A/D (analog to digital) converter,

so its analog input pin can read the analog signal from the sensor directly. It also can convert the analog signal to binary integer from 0 to 1024.[12]

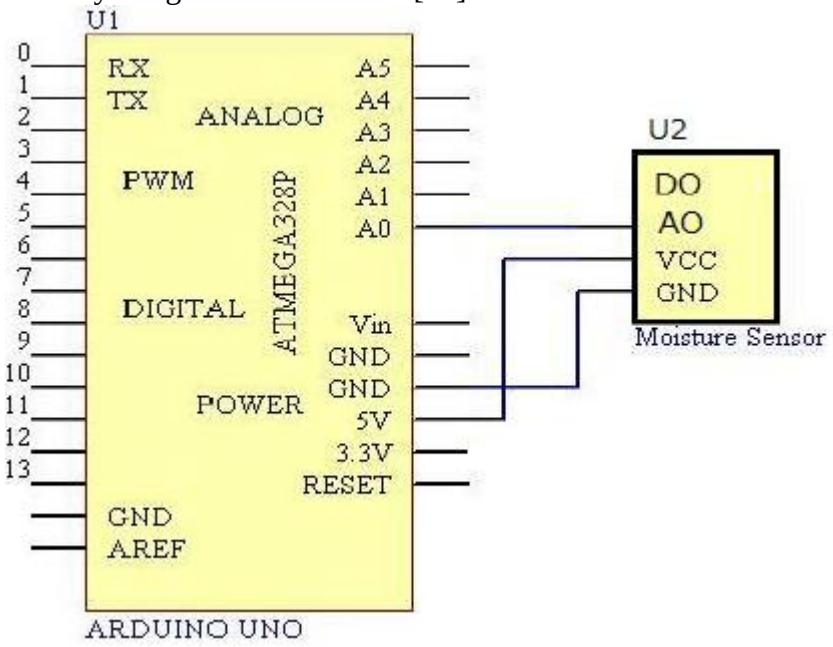


Figure [9]: Connection between Arduino UNO and Sensor

2.1.2 ELECTRICAL CIRCUITRY DESIGN

This section will demonstrate the features of different components in the E-Garden in three different stages, which are proof of concept, prototype and production. Also, some further improvements for the system will be mentioned in the following article.

2.1.1.1 WIRELESS SENSOR CIRCUITRY

In order to achieve the wireless feature for the sensors, we built a circuit with Arduino controller and Bluetooth 4.0 module. The reason we choose Bluetooth 4.0 is because Bluetooth 4.0 has less power consumption and it is also suitable for long distance connection. The figure [10] shown below are the rough design of the circuit in prototype stage.

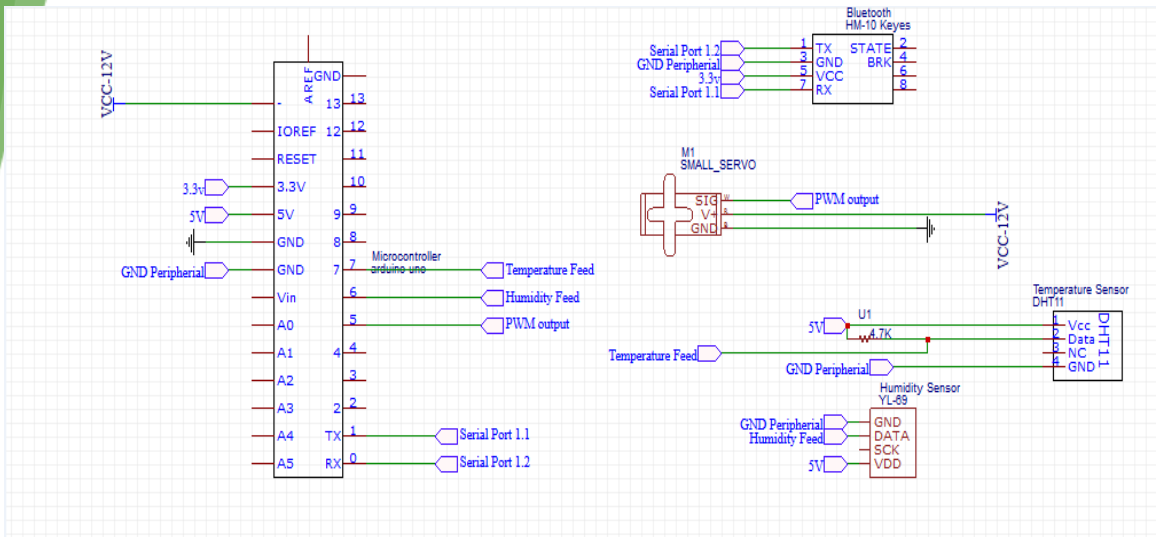


Figure [10] Schematic diagram of the electronic system

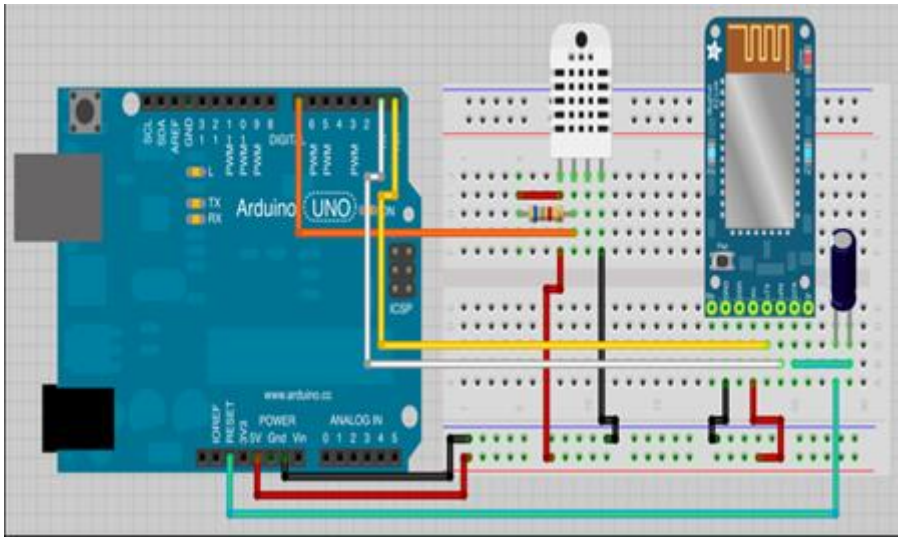


Figure [11] Bluetooth module and sensors connection

In this circumstance, the data collected by the sensors will transfer to the Arduino through wire and Arduino will send the data to the main Raspberry Pi 2 controller through Bluetooth 4.0. In this stage, breadboard is used to test the circuit's work ability. There are still some adjustments that can make the circuit work better. For example, capacitors can be added in order to lower the noises that may affect the data transfer. However, in the production stage, all the components should be soldered together and directly connected with each other in a safe enclosure.

2.1.1.1 WATERING SYSTEM CIRCUITRY

In order to achieve the auto watering system, we designed to control the water tap that can turn on / off the water system. Therefore, we tied the servo-motor on the water tap and make sure the water is available to come out and have enough force. In the prototype stage, we connected the servo-motor directly with the water tap and tied it on the water pipe. Since the servo-motor controlled by other micro-controller "Arduino", the micro-controller can receive data from the raspberry pi 2 by using Bluetooth. Therefore, the controller can command the motor on/ off, as we set that 0 is close and 1 is open in the code. Since we had measured the water tap in circle around 2.5-3.0 cm in radius, and the water amount is around 100mL/s.

Time spend(s)	Amount of water(mL)
1	98.7
2	191.4
3	296.2
4	398.3
5	501.8

In the production stage, we consider that using an irrigation system to control the output figure [12], so that we accurately measure the amount of water and watering time. The graph shown below is the ideal irrigation system with our E-Garden installed.

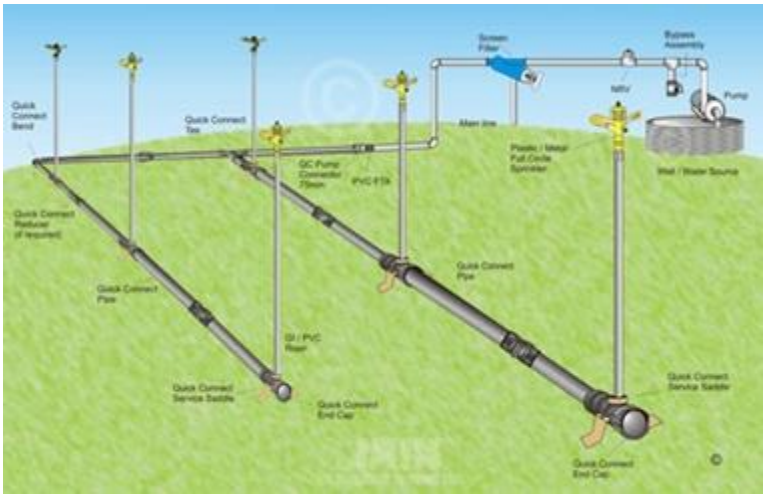


Figure [12] Ideal irrigation system apply our E-Garden system

2.1.3 POWER SUPPLY DESIGN

For the E-Garden, there are five main components that have to be powered. They are the main controller-Raspberry Pi 2, the sub-controller- Arduino UNO, servo, temperature sensor and humidity sensor. The table below will demonstrate the electrical characteristic of these components.

Raspberry Pi 2	5 Volt DC
Arduino UNO	5-12 Volt DC
Servo	5 Volt DC
Temperature sensor	5 Volt DC
Humidity sensor	5 Volt DC

Table [2] Data of hardware parts

As the previous part mentioned Arduino UNO is powered by USB port in the prototype stage. However, for the real product, we can directly plug the power supply cable into the socket or use an AA battery pack to power the Arduino UNO, just like the graph show below.

Figure [13] Appropriate method of powering Arduino

For the further improvement, rechargeable Li-ion battery figure [14] is also a good choice because it is safer and it has less environmental pollution than the normal battery. Also, rechargeable Li-ion battery has longer sustainability compare to the normal battery.

Figure [14] Alternative method for powering Arduino

2.1.4 ENCLOSURE DESIGN

While designing the enclosure, the following requirements for the material have to be considered:

1. As the sensor part works outside, the enclosure has to be waterproof and insulated to protect the electronic components inside.
2. The enclosure has to be firm enough and lightweight to prevent the harsh weather condition and movement damage.

Therefore, the ABS plastic is the best choice for the material.

Despite the two requirements mentioned above, another requirement of the design is that the enclosure should be portable. Specifically, this means the case should accommodate all of the components: the Arduino UNO, the TL-69 moisture sensor, the Bluetooth module and the power supply.

The following table [3] shows the sizes of the interior components.

	Arduino UNO	TL-69 sensor	Bluetooth module	Power supply
Shape	Rectangle	Rectangle	Rectangle	Rectangle
Quantity	1	1	1	3
Dimension (length*width)	6.86cm *5.34cm [13]	1.5cm *1.5cm [14]	2.69cm*1.3cm *0.22cm[15]	3*5.50cm*1.50cm [16]

According to these data, we construct a case with the components inside as shown in figure [15]. The length of the case is 13cm, the width is 11cm, and the height is 2cm. The thickness of each side is 0.5cm. In the left of the case, there are two series holes. The first two holes are for the connection between the YL-69 sensor and its probes; the other four holes are for the connection between the DHT11 sensor and the Arduino UNO.

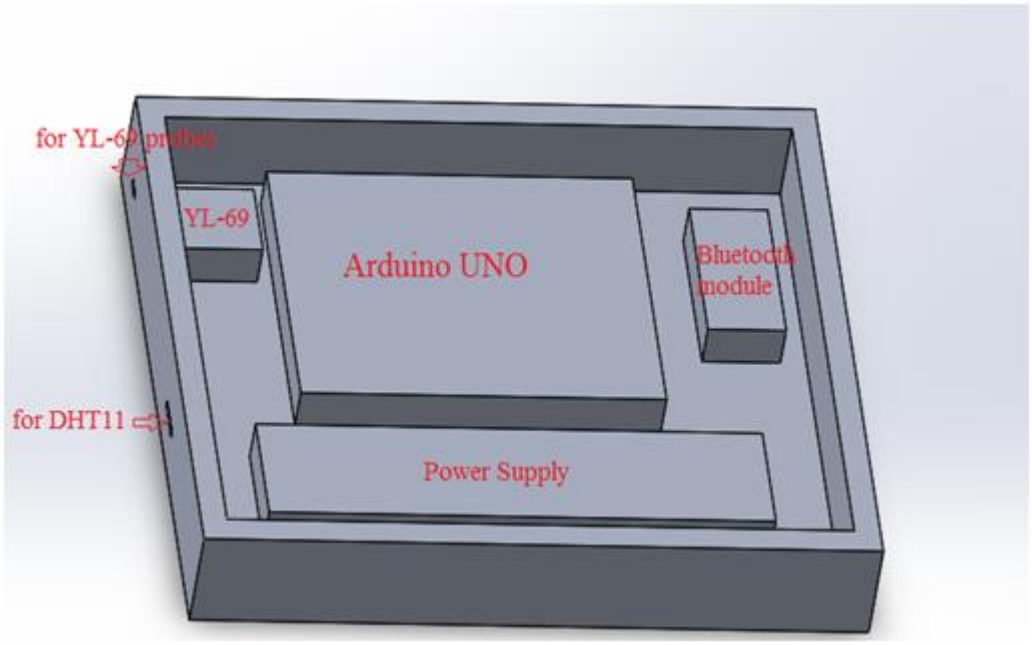


Figure [15]: the case and the interior components

The next significant design part is that the holes which on the case have to be waterproof. The gaps between the connection wires and the holes are as small as possible. We will use the rubber O-ring to fill these gaps, to prevent the water seepage.

2.2 FIRMWARE DESIGN

The firmware section describes the program running on the raspberry pi. It mainly contains two major features which are Wi-Fi connection and Bluetooth connection. In this following section, we will explain how Raspberry Pi is setup and how Wi-Fi and Bluetooth connection works.

2.2.1 RASPBERRY PI

Raspberry Pi is the main micro-controller of our product. It acts like a bridge between web app and other peripheral devices. The raspberry pi is used for receiving data from webserver and the other microcontrollers - "Arduino". The raspberry pi is a mini-computer that can be easily set up with Wi-Fi network and Bluetooth network by using USB adapters. The graph below figure [16] has shown that raspberry pi has received data from Bluetooth, and then the data will be saved into the MySQL database. From MySQL, we are able to send the data out to the web server. On the other hand, users can also send signal from the web app to the raspberry pi and it will send feedback to servo-motor to give us output. Figure [17]

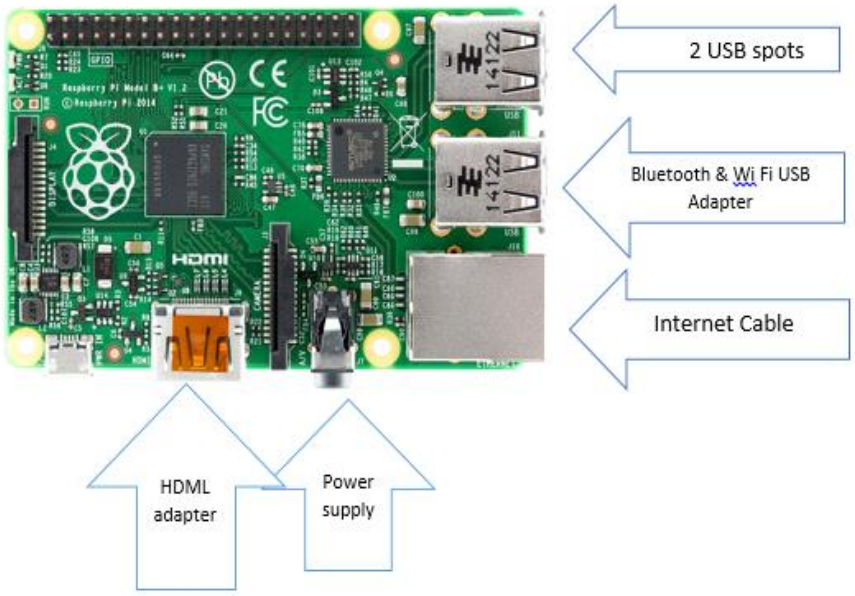


Figure [16]: Raspberry Pi basic structure

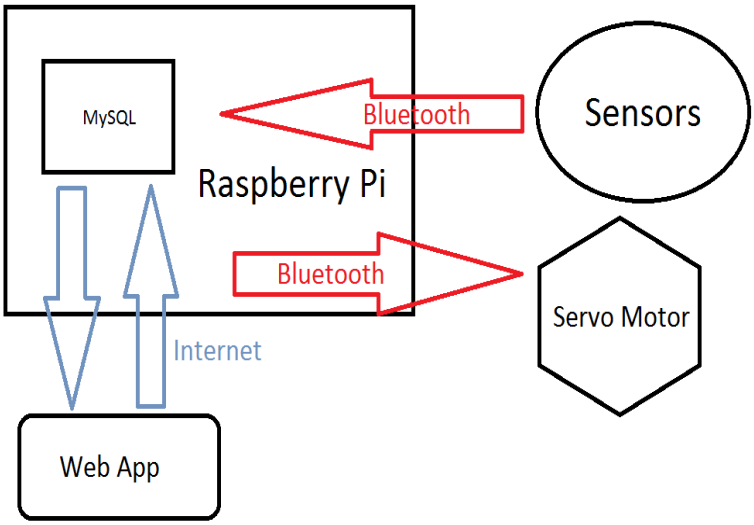


Figure [17]: The communication between Raspberry Pi and other devices in different ways

2.2.2 RASPBERRY PI SETUP

Raspberry is a small computer, so we have to setup a Wi-Fi and Bluetooth inside the machine. We consider that there are two ways to set up the network.

- We have to have a mouse, keyboard and a screen that require a HDMI adapter that we can easily setup the internet like normal computer.
- On the other hand, we can just use an internet cable that can connect the raspberry pi without anything, but only plug in the cable.

2.2.3 DEVICE COMMUNICATION

2.2.3.1 WI-FI CONNECTION

Raspberry pi is a micro-controller and it is the main part of the project. We have chosen to use Linux Operated System (OS) in the microcontroller, so that we can use terminal to setup the communication between the network and microcontroller. Since the controller has provided the USB for us to connect the mini USB wireless adapter, we just have to setup the command on terminal and make sure the IP address connect to the mini USB wireless adapter. By using a 150MB per second wireless adapter, we can easily send out data to the web server and reverse data from the internet in a high speed. From the graph below Figure [18], we can clearly see how the raspberry pi communicates with the web server.

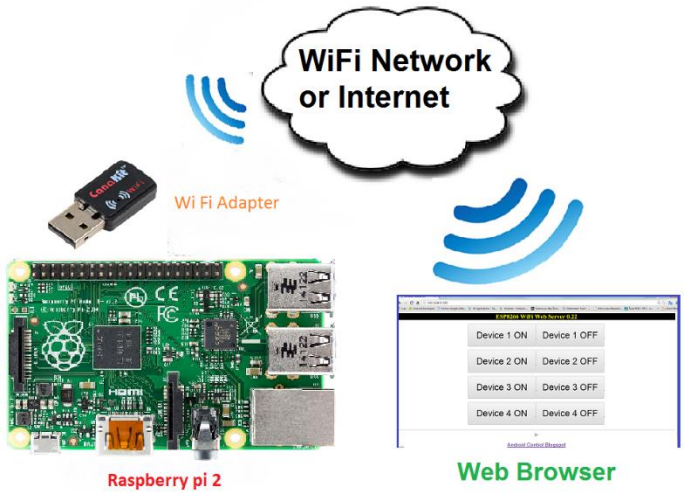


Figure [18]: The communication between Raspberry Pi and network

2.2.3.2 BLUETOOTH CONNECTION

Same as Wi-Fi connection, we have chosen a Bluetooth USB adapter that has low power consumption, 3 MB per second data rate and a receiving/sending range in 20-50m which can easily setup the communication with temperature sensors, moisture sensors and servo motor. As those sensors and motor has to be using outside the house, so that we had to use Bluetooth sensors to minimize the power. Since the data cannot be send it to internet directly, so that we had design the microcontroller as the platform that the data can be communicate with the server/users' application.

2.2.3.2 BLUETOOTH SETUP

In Bluetooth setup we have to check the Bluetooth address of the device USB adapter on the line BD address from the terminal after typing "sudo hciconfig". Afterward we can use the terminal on the Raspberry Pi to scan the Bluetooth device that nearby and collect the address of the sensors and motor to connect them together. Since we are collecting a set of number data, we can save those data under a text file and send it to the database "MySQL" inside the raspberry pi.

2.3 SOFTWARE DESIGN

As mentioned in the previous sections, AWS was chosen to host the web APP for E-Garden, thus it supports all platforms with internet access. This decision fulfills the requirement [Req 3.3.3-P1]. Elastic Beanstalk is a PaaS offered from AWS which allows the users to launch web app and push them to a definable set of AWS services including Amazon EC2, Amazon S3 and Amazon Simple Notification Service. The Elastic Beanstalk is in charge of the capacity provisioning, load balancing, scaling and application health monitoring. For the proof-of-concept model, only the data for one user is illustrated. However, the production model will potentially have a large number of users accessing their plants information with their own username and password. Thus, one of the reasons behind choosing AWS and Elastic Beanstalk to host the web app is that they make load balance and big data analyze much easier. User data will become a valuable resource in the future. AWS has the function that can easily analyze the data when our product comes to the production stage.

2.3.1 HOME PAGE DESIGN

The home page contains the information of the plants, which is the moisture level and the soil temperature. The watering modes are also controlled on the homepage, where the users can choose between automatic watering and manual watering. With auto-watering turned on, the system will water the plant when necessary. Manual watering will water the plant if the user presses the water button. It fulfills the requirement [Req 3.3.4-P2] for the web app.

2.3.1.1 USER INTERFACE DESIGN

The homepage of the web app is the primary interface between the system and the user. The design of the User Interface should fulfill the requirement [Req 3.3.5-P1] and [Req 3.3.6-P1]. The soil temperature and soil moisture level are displayed on the main page. In addition, there are two buttons at the bottom of the homepage. The top button is labeled "Auto Watering On/Off", and the bottom button is labeled "Water Plant Now!". When the user presses the top button, the automatic watering will be turned on or off. When the other button is pressed, the web app will send a one-time watering request to the controller. The user interface of the web app is shown in Figure [19].

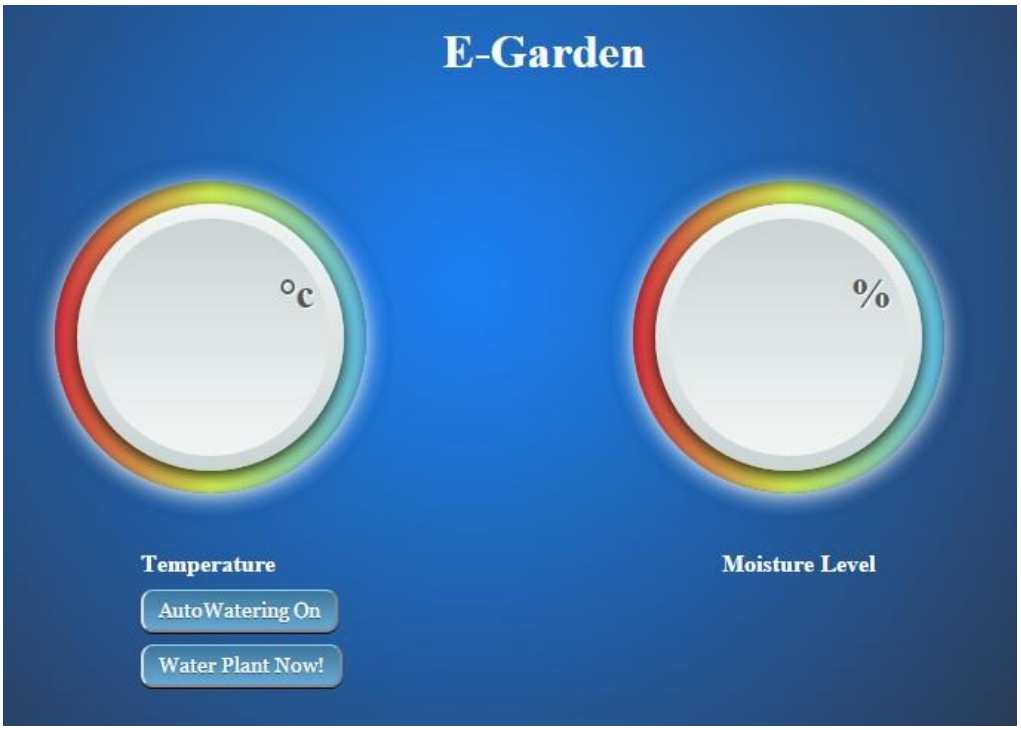


Figure [19] Homepage UI Design Demonstration

2.3.1.2 DATABASE DESIGN

The database is deployed on AWS by Amazon RDS (Rational Database). The database environment is set to MySQL, which makes it easier to handle the JSON file sent by the micro-controller. When the web app receives a JSON object, it decodes the JSON file and insert the data into MySQL database. The database creates tables to store all data and user information.

2.3.1.3 AUTO-WATERING FUNCTION DESIGN

Since the soil temperature and moisture level would not change dramatically in a short time, the information is refreshed every 3 hours. If auto-watering is turned on, the python script will check if the moisture level is lower than the threshold every time it receives the JSON file. When the moisture level drops below the threshold, the web app will send a request to the microcontroller to water the plant. The flowchart of the check process is demonstrated below in Figure [20].

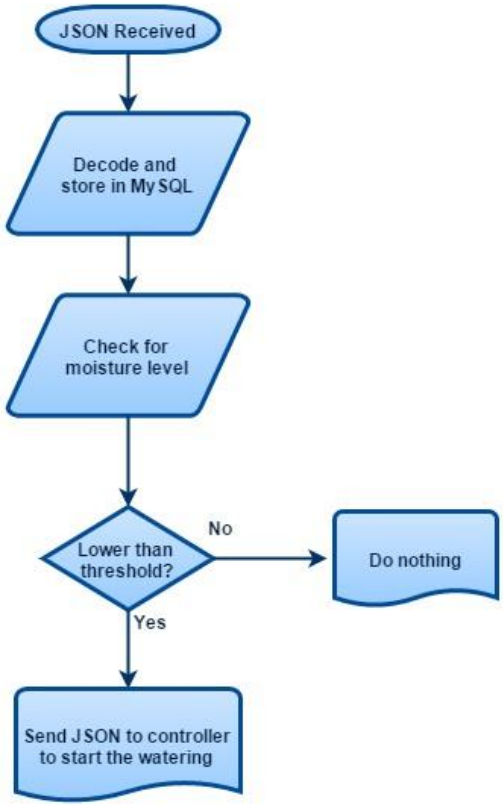


Figure [20]: Flowchart of Auto-watering Function

2.3.2 LOGIN FUNCTION DESIGN

In our software development, the login system is included in our web server. The login function will include the following:

- username,
- password
- email
- registration

Clients are able to login our web server with their unique registered information, such as username, password, and email. The login system will use the shared database in our web server. With more and more stories of cracking in the user information, we also add a secure function in our login system. The schematic design process is shown in figure [21].

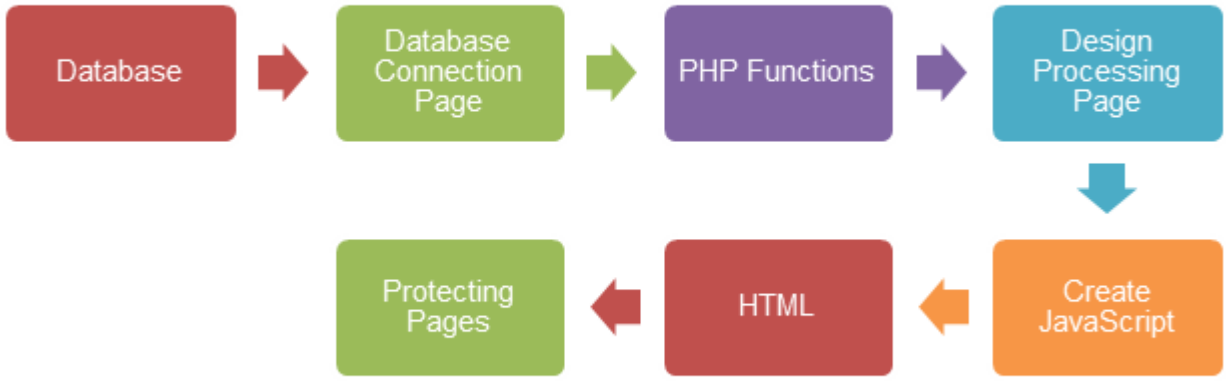


Figure [21]: Design Scheme for the Login Function

The detailed design information for each component on Figure [21] is following:

a) Database

In order to setup your database, the following PHP and MySQL need to install on your system:

- PHP version 5.3 or later
- MySQL version 4.1.3 or later

In this section, MySQL database and table name are created to store login information. Moreover, we create a user with only SELECT, UPDATE and INSERT privileges. The reason for this is that if there was ever a breach of security in our script the hacker couldn't delete or drop anything from our database.

b) Database Connection Page

A PHP code will be created here to connect our database. Moreover, for security purpose, a file contains global configuration variables will create outside the web server's document root. So, if someone mistakenly dropped the 'php' extension, say, or messed up the file permissions the file still could not be displayed as text in a browser window. [2]

c) PHP Functions

In this section, PHP functions will do all the processing of login script. The main functions for PHP are:

- Create the login function
- Check logged in status
- Sanitize URL

For the security concern, the following functions were added to enhance the security of our login system:

- Add security access function in each PHP session
- The Brute Force Function [3]

d) Create processing pages

The following main functions will be created at here:

- Login processing page
- A logout script
- Registration page

e) Create Javascript Files

Two main Javascript files will be built at here

- Sha512.js file
  It uses the hashing function so our passwords don't get sent in plain text.
- Form.js file
  It will handle the hashing of the passwords for the login and registration forms.

f) HTML Pages

In this section, HTML pages will be created to display the following information:

- Login form with "email" or "username", and "password"
- Register success page
- Error page

g) Protecting Pages

A page protection script will be created to check if the user is logged in.

2.3.3 HITORIC DATA DISPLAY DESIGN

In our web app, historic data is displayed in a separate web page. In that web page, we will design a history table to record the watering history within last nine days and current date. The purpose of history table is that to give user a clear and intuitive way to view. We are going to create HTML form web page with dynamically data using JavaScript & HTML Tables and PHP code to process them in the back end.

We make the webpage that can:

The added data can be removed automatically within the set limits.

PHP script to fetch the data from our database.

The following flowchart shows the design process:

1. Connect to Database

The first thing to do is connect to the database. We use the function “mysql_connect” to connect to MySQL. This function returns a resource that is a pointer to the database connection.

2. Filtering data

Selecting about what data to return from the database. Sorting results by Date and watering status, and choose the number of rows to return.

3. Set limit

Our history table design to store the latest ten data. The newest data will replace the oldest one.

4. Retrieve data

The retrieve function returns all of the data as an associative array.

5. Displaying data

Build the table outputted by “display_as_html_table”, including changing the order of the fields, the table headings or labels.

Our history webpage will look like this:

DATE	STATUS	
Monday, November 9, 2015	Yes	
Tuesday, November 10, 2015	Yes	
Wednesday, November 11, 2015		No
Today	Yes	
… …		

3. Test Plan

3.1 INDIVIDUAL COMPONENT TEST

3.1.3 SOFTWARE TEST PLAN

The software part of the project consists of a web app and a server database to store the information acquired from the sensors. The web app is the interface between user and the system, thus the UI must be simple and easy to use. The web app has three main web pages, which are the login page, the homepage, and the history data page. The three web pages are individual pages with the login page linked to the homepage and the homepage linked to the history data page. All three web pages manages communication to the database through separate python scripts. Therefore, the three pages will be tested as individual parts for the following tasks:

3.1.3.1 HOMEPAGE TEST PLAN

- Successfully decodes and stores the JSON file sent by the controller.
- The correct soil temperature and moisture level are displayed on the homepage.
- The soil temperature and moisture level are renewed every time a JSON from controller is received by the web app.
- Pressing the "Auto Water" button enables the check function to check the moisture level every time a JSON is received from the controller.
- If the moisture level is lower than the threshold, a JSON is sent to the controller.
- Pressing the "Water Plant Now!" button sends a JSON to the controller.

3. 2 INTEGRATION TESTS

The integrated system will be tested after the individual components test has completed. This test will mainly focus on the communication between different parts of the project to make sure the integration is successful. A system test will be done to test the complete functionality of the system. Power consumption of the system is measured during power consumption test. Last but not least, a stress test is performed to ensure the reliability of the system.

3.2.1 HARDWARE-FIRMWARE INTEGRATION

- YL-69 sensor collects the soil moisture successfully.
- DHT11 sensor collects the temperature sensor successfully.
- After successfully collecting the moisture and temperature data, the sensors can send the data to the Arduino UNO successfully.
- The unit tests will be conducted by connect sensors to Adruino Uno independently to check if desired output is obtained for specified input, and the output will be observed from COM monitor in Arduino IDE, and the input such as

temperature and humidity will be collected from environment and use thermometer and hygrometer.

Component Name	Test Plan	Results
YL-69 Humidity Sensor	Collect sample data of humidity at SFU campus at 6:00AM, 2:00PM and 2:00AM from both YL-69 and hydrometer	The error between data from YL-69 and hydrometer should not exceed 10%
DHT11 Temperature Sensor	Collect sample data of temperature at SFU campus at 6:00AM, 2:00PM and 2:00AM from both DHT11 and thermometer	The error between data from DHT11 and thermometer should not exceed 10%
Servo Motor	With Servo library from Arduino community, observe and record position data	The error between present position in Arduino program and the actual position should not exceed 10%
Bluetooth Module	With Bluetooth 4.0 library from Arduino community,	

3.2.2 SOFTWARE INTEGRATION

- The AWS RDS is accessible to all three web pages.
- After successfully login to their accounts, customers are directed to the correct homepage.
- The history data link on homepage directs to the correct history data web page.
- History data displays the correct number from database.

3.2.3 SYSTEM INTEGRATION

- The web app displays the right temperature and soil moisture level acquired from sensors.
- The auto-watering function is watering the plant if soil the moisture level drops below the threshold when the controller sends the plant information to the web app.

- Pressing the “Water Plant Now!” button on the web app will make the watering system work immediately.

Test case	Test method	Test result
Turn autowatering on when outside condition is satisfied	• Check if water is released by servo motor. • Use measuring cup to contain water released	Water is supposed to be released, and the error between realised water amount and pre-set water amount should not exceed 20%
Click “water plant now” when outside condition is satisfied	• Check if water is released by servo motor. • If water released, Use measuring cup to contain released water	Water is supposed to be released, and the error between realised water amount and pre-set water amount should not exceed 20%
Click “autowatering on” when outside condition is not satisfied	• Check if water is released by servo motor. • If water released, Use measuring cup to contain released water	Water is not supposed to be released, and the error between realised water amount and pre-set water amount should not exceed 20%
Click “water plant now” when outside condition is not satisfied	• Check if water is released by servo motor. • If water released, Use measuring cup to contain released water	Water is supposed to be released, and the error between realised water amount and pre-set water amount should not exceed 20%

3.3 POWER CONSUMPTION TEST

- Determine the average and peak power consumption of each hardware components.
- Determine the average and peak power consumption (to be written as a specification in the user manual) of whole system.

3.4 STRESS TEST

•Run the system over 24 hours for any crash or error to test its reliability.

4. Conclusion

The design specification provides the design specifications and technical details which correspond to the functional specifications of the E-Garden. The system consists of three major sections:

● Hardware development aimed to provide a real time feedback system

● Firmware development on the microcontroller to achieve digital packages transmission through Wi-Fi and Bluetooth network to multiple sensors

● Software development aimed to develop a web server to control the E-Garden

All of the components above had their own section explaining the specifications and justifications during the design phase. The final section of the document provides a set of preliminary test procedures to examine the model functionality of the proof-of-concept model.

5. References

[1] [2]: No author, "DHT11 Humidity & Temperature Sensor", 2010. [Onlie]. Available: http://files.amperka.ru/datasheets/dht11.pdf. [Accessed: November 12, 2015].

[3] [4] [5] [6]: No author, "Temperature and Humidity module DHT11 Product Manual", 2010. [Onlie]. Available: http://akizukidenshi.com/download/ds/aosong/DHT11.pdf. [Accessed: November 12, 2015].

[7]: Brent Q. Mecham, "A practical guide to using soil moisture sensors to control landscape irrigation", 2013. [Onlie]. Available: https://watergreat.com/reference/Practical_overview_2010.pdf, [Accessed: November 12, 2015].

[8] [9] [10]: No author, "Soil Hygrometer Detection Module Soil Moisture Sensor for Arduino Smart Car", 2011. [Onlie]. Available: http://www.ebay.co.uk/itm/Soil-Hygrometer-Detection-Module-Soil-MoistureSensor-For-arduino-

Smartcar/230947908317?hash=item35c5916add&pt=UK_BOI_Electrical_Components_Supplies_ET, [Accessed: November 12, 2015].

[11] [14]: No author, “YL-69 Moisture Sensor Schematic”, 2013. [Onlie]. Available: http://m2.img.dxcdn.com/CDDriver/sku.200142.jpg, [Accessed: November 12, 2015].

[12]: No author, “Arduino Uno”, 2012. [Onlie]. Available: http://arduino.cc/en/Reference/AnalogPins, [Accessed: November 12, 2015].

[13]: No author, “Arduino UNO & Genuino UNO”. [Onlie]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno, [Accessed: November 12, 2015].

[15]: No author, “BLUETOOTH 4.0 BLE MODULE”, 2015. ”. [Onlie]. Available: http://leeselectronic.com/product/15545.html?search_query=bluetooth+4.0&results=10, [Accessed: November 12, 2015].

[16]: No author, “ENERGIZER E91”, 2009. ”. [Onlie]. Available: http://data.energizer.com/PDFs/E91.pdf, [Accessed: November 12, 2015].