Water Smarter Information

Who are we?
Group of college students and recent graduates passionate about sustainability

What are we doing?
Creating a modular system that automatically water plants based on local weather station data, regional time, and live sensor data collection. An algorithm that will correctly maintain the green roof after planting

Where do we want to go in the future?
We will create a sensor/code based system that will show a user what food can be best grown in their geographic location. In other words, communicate the feasibility of specific plant species for optimal growth in the region

Who will benefit from this?
- facilitate/reduce the amount of work needed for garden maintenance?
- Water saving

Slide Deck: Pitch Deck
Mural: Mural board

Misc Research
“Smart Products Manufacturers of irrigation controllers in particular are using technology to customize watering regimens based on homeowners’ needs, location, and weather. Smart controllers have been on the market since 2014 and are designed to make watering more efficient based on weather and soil conditions for a particular home. Recent new product introductions have integrated smart technologies to enable consumers to control watering systems remotely with more customization and water-saving options. Lawn & Garden Watering Products: United States ©2019 The Freedonia Group. All rights reserved. Market Environment 8 Companies outside the irrigation industry began to launch inexpensive, Wi-Fi-enabled irrigation controllers in about 2014. Rachio, Skydrop, and others marketed devices directly to consumers. Irrigation product manufacturers responded to outside competition with Wi-Fi enabled smart controllers of their own. Orbit Irrigation’s smart Wi-Fi controller bhyve, launched in 2016, is designed to save water and can be controlled from a smartphone, tablet, or computer. “ -- **Water Supply and Irrigation Systems: Global Markets to 2022 BBC market report**

Table of Contents

**Indicates executable code*

Demo Video

GOAL: tell a story of why we chose this project, what and how it works.

Content

- Show circuit breakdown
- Demonstration-show all windows at same time. Can be filmed separately
    - Solenoid valve
    - Moisture sensor
    - Values changing in arduino ide

Project Features

- Solenoid valve controlled by moisture sensor value
- RPI convert arduino sensor value to csv file
- Python web server to display value
    - Screen record

Lights, Camera, Action

Prep Work

- White background=done
- Tripod=done

Content

- Solenoid opening and closing in sink
    - 3d printed pieces for extension
- Soil moisture sensor
    - Water
    - air
    - Dry soil
    - Watering being poured in

Video Outline

1. Pan over hardware. Wipe down the slides title slide
2. Introduction (WHY for the project)
   Present a need for our product and how they relate ?
   a. Hook=use the short description in the devpost

3. Identify components (pan to a corner after showing)
   a. Arduino
   b. Soil Moisture
   c. Solenoid Valve
   d. RPI

Through the use of the internet of things, we can automate. Here, we have an arduino, microcontroller, which connects to sensors in order to collect data. One sensor we explored is the soil moisture sensor which sends an analogue value relatively displaying the level of moisture in the medium of contact. Then, there is the solenoid valve where the microcontroller will send current in order to open or close the valve.

4. Demonstration (fuzz to videos)
   a. Show each component in action for 15 seconds?
   b. Narrate the following
      i. What is happening
      ii. If statements, etc. conditions

We preliminarily experimented with the internet of things and quickly prototyped a product where the microcontroller will open the valve if the soil is dry or close to the value given if exposed to air until an optimal value of moist. Then, close if reached a saturation or the value given if exposed to tap water. With the data, we have created a simple web server which displays the current value of the system: either open or closed, as well as send the values to the Raspberry PI who saves the values as a csv.

5. Impact, applications (wipe down)
   a. Why Water Smarter?
   b. Scalability
   c. Future Plans
6. Catchy ending
7. Contact Slide

https://youtu.be/hWTPxe32pa0

Promotional Video

GOAL: boomerang video promoting the product

1. Water flow
2. Wiring Timelapse
3.

Moisture Sensor Information

315	Water^	
X	Soil	
900	Air^	

		%Moisture
	315	100%
585	432	80%
117	549	60%
	666	40%
	783	20%
	900	0%

	Min	Max
Water	221.92	307.17
Soil	348.58	895.79
Air	901.22	914.1

TESTED IN SOIL WITH WATER ADDED

The moisture sensor we are using has both analog and digital value output. The digital value is a boolean of 1 or 0 which indicates whether or not to dispense water. The analog value gives an arbitrary range which we identified through trials to find an upper and lower bound. The upper bound was found through keeping the sensor dry and in contact with the ambient air as the only fluid. The lower found was found through keeping the sensor submerged in tap water as the only fluid. Once we identified this range, there were approximately 585 numerical values in between which were broken into increments of 117 or 20% of 585 which relatively gives the %moisture of the material or fluid the sensor is in contact with. The range has saturation as the lower bound and dry as the upper bound.

With this new range of sensor values, we created a conditional statement for the arduino to control the solenoid valve. If the value is below the lower bound, the microcontroller will turn the solenoid valve off; however, turn on if above the upper bounds. This action will take place until the moisture sensor is within the defined range.

In the next step, we are creating a user interface where they can either (1) choose a relative moisture range as defined above or (2) choose a select plant which contains a specific relative moisture range for optimal growth.

Arduino Code

Moisture Sensor

```
// Link

#define Moisture A0 //The definition of AO pin IO-A0
#define DO 7        //The definition of DO pin IO-7

void setup() {
  pinMode(Moisture, INPUT); //Define A0 as input mode
  pinMode(DO, INPUT);
  Serial.begin(9600);
}

void loop() {
  //Returns the serial measurement data
  Serial.print("Moisture=");
  Serial.print(analogRead(Moisture));//Numerical read AO
  Serial.print("|DO=");
  Serial.println(digitalRead(DO));//Numerical read DO
  delay(1000);
}
```

Solenoid Valve

```
int solenoidPin = 4;    //This is the output pin on the Arduino we are using

void setup() {
  // put your setup code here, to run once:
  pinMode(solenoidPin, OUTPUT);          //Sets the pin as an output
  Serial.begin(9600);
}

void loop() {
  //Returns the serial measurement data

  digitalWrite(solenoidPin, HIGH);    //Switch Solenoid ON
  Serial.println("ON");
  delay(10000);                    //Wait 1 Second

  digitalWrite(solenoidPin, LOW);     //Switch Solenoid OFF
  Serial.println("OFF");
  delay(10000);                    //Wait 1 Second
}
```

*Main Code

```
int solenoidPin = 4;    //This is the output pin on the Arduino we are using
#define Moisture A0 //The definition of AO pin IO-A0
#define DO 7        //The definition of DO pin IO-7

void setup() {
  // put your setup code here, to run once:
  pinMode(Moisture, INPUT);//Define A0 as input mode
  pinMode(DO, INPUT);
  pinMode(solenoidPin, OUTPUT);          //Sets the pin as an output
  Serial.begin(9600);
  Serial.println("  Moist?");
}

void loop() {
  if(analogRead(Moisture) < 666){
     Serial.println("0");
   } else if(analogRead(Moisture) <= 315){
     Serial.println("-");
   } else {
     Serial.println("1");
     }

  if((analogRead(Moisture)) < 310){
    //Turn solenoid valve off
    digitalWrite(solenoidPin, LOW);     //Switch Solenoid OFF
    //Serial.println("Solenoid Valve is OFF");
    delay(5000);                    //Wait 5 Second
  }

  if((analogRead(Moisture)) > 900){
    //Turn solenoid valve on
    digitalWrite(solenoidPin, HIGH);     //Switch Solenoid OFF
    //Serial.println("Solenoid Valve is ON");
    delay(5000);                    //Wait 5 Second
  }

  if((analogRead(Moisture)) < 900 && (analogRead(Moisture)) > 310){
    //Turn solenoid valve on
    digitalWrite(solenoidPin, HIGH);     //Switch Solenoid OFF
    //Serial.println("Solenoid Valve is ON");
    delay(5000);                    //Wait 5 Second
  }

}
```

CSV Style Print Data

```
#define Moisture A0 //The definition of AO pin IO-A0
#define DO 7        //The definition of DO pin IO-7

void setup() {
  pinMode(Moisture, INPUT); //Define A0 as input mode
  pinMode(DO, INPUT);
  Serial.begin(9600);
}

void loop() {
    String value = String(analogRead(Moisture));
    Serial.print(value + ",");//Numerical read AO
    delay(1000);

}
```

Raspberry PI Code

1. Install Arduino through terminal

sudo apt-get install arduino

2. Install Arduino Serial through terminal

sudo apt-get install -y python-serial

3. Upload Arduino Code

n/a

*4. Acquire Arduino Data and Save as CSV

\\LINK

```
import serial
import re
#import time

arduino_port = "/dev/ttyACM1" #serial port of Arduino
baud = 9600 #arduino uno runs at 9600 baud
fileName="analog-data.csv" #name of the CSV file generated

ser = serial.Serial(arduino_port, baud)
print("Connected to Arduino port:" + arduino_port)
file = open(fileName, "a") #appends
print("Created file")

while True:
    getData=str(ser.readline())
    data=getData[0:][:-2]

    data2 = re.findall('\d', data)
    print(data2[0]);

    file = open(fileName, "a")
    file.write(data2[0]) #write data with a newline
```

Add-On Raspberry PI Code

*1. Python Web Server

https://projects.raspberrypi.org/en/projects/python-web-server-with-flask/3

```
from flask import Flask
import serial
import re

arduino_port = "/dev/ttyACM1" #serial port of Arduino
baud = 9600 #arduino uno runs at 9600 baud

ser = serial.Serial(arduino_port, baud)
getData=str(ser.readline())
data=getData[0:][:-2]

data2 = re.findall('\d', data)
str2 = str(data2[0])
print(data2[0]);

app = Flask(__name__)
@app.route('/')

def index():
  str = "Smarter Water "+ str2
  return str

if __name__ == '__main__':
  app.run(debug=False, host='0.0.0.0')
```

2. Python Firebase Connection

2A. Install the Firebase package

$ sudo pip install python-firebase

$ sudo pip3 install adafruit-blinka

$ sudo pip3 install adafruit-circuitpython-mlx90614

$ sudo pip3 install pyrebase

2B. N/A

NOT FINDING FIREBASE KEYWORD/VARIABLE=ISSUE

X. Setting up a Raspberry Pi as a routed wireless access point

https://www.raspberrypi.org/documentation/configuration/wireless/access-point-routed.md

Accessing OpenWeatherMap API

Tutorial link:
https://techtutorialsx.com/2018/03/17/esp32-arduino-getting-weather-data-from-api/
Example:
https://www.instructables.com/ESP32-WiFi-Weather-Station-With-a-BME280-Sensor/

Parsing JSON in Arduino

ArduinoJSON (not Arduino_JSON): https://arduinojson.org/