2020

ESSTECH Challenge 2020
Human Computer Interaction

[CONCEPT AND DESIGN]

TEAM LF

Directory

1. The Concept

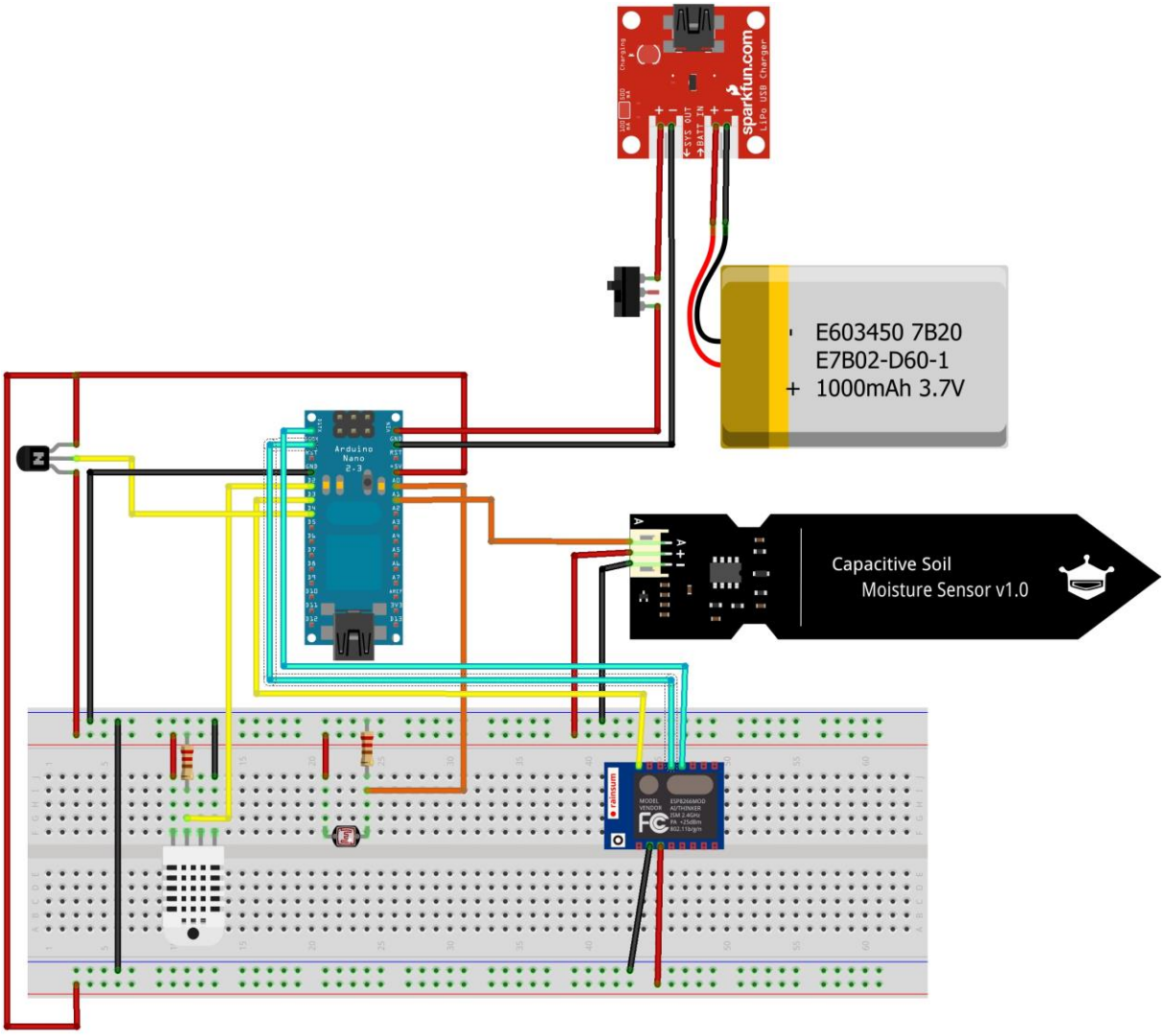
We decided to develop a system which would help the customer to learn about the basic needs of different plants by providing a small encyclopaedia and monitoring the plant's current state.
Within this context we thought about which parameters of the plant growth could be measured and effectively influenced. These turned out to be humidity, temperature, light intensity and soil moisture gradient. By measuring these parameters and sending them to the customers phone, our system will enable them to have a nice, green living space without having to learn about Botany or worrying if they are neglecting their plant because of a busy schedule. This will be optimal for private households with pot-plants to conservatories.

But our product can also be used in small florists' businesses to be able to precisely tend to delicate flowers like lilies, thus decreasing the risk of a financial loss.

Because our target customers will primarily be the individual our system has to be very simple. It basically is made up out of two parts: The sensor-array which will be planted in the soil near the chosen plant and an app. The App is connected to the sensor-array by a wifi-modul and will display the gathered data. The customer will be able to choose the monitored plant out of a catalogue within the App the give a reference point for the data.

2. The Prototype

2.1 Hardware



Circuit Diagram

Our system is made up out of the following components:

- An Arduino Nano Board as the main connection hub
- A Capacitive Soil Moisture Sensor
- A temperature and humidity sensor
- A photoresistor
- A wifi-modul
- An USB Lithium Polymer Charger with a voltage stepup-converter
- On-Off switch
- NPN-Transistor
- Resistors

To be able to estimate a price point we researched the average price for each component. We came up with:

Name	Average B2C Retail Price [Euro]
Arduino Nano V2.3	3
DHT11 Humidity Temperature Sensor	4,5
Photoresistor	1
Capacative Soil Moisture Sensor	3
Lithium Polymer Battery	8
USB LiPoly Charger	16
Cable, Switch, Resistor, NPN- Transistor	3
Wifi-Modul	6
	44,5

Since we are talking about the retail price for the individual customer the price seems rather high. But if we were to buy the components directly from the manufacturer our estimated price could drop by about quarter to a third.

2.2 Software

The code is written in Arduino IDE as development tool. As typical for Arduino the code is divided in the initialising section, the setup section, the loop section and a few functions that are called during the loop. After initialising all the variables and pins our sensors and other electronics are wired to, the setup sets the WIFI-module either as access-point for first time configuration, to enable a connection on which the user can submit the needed data for router access and API key (personal key for online database storage of the sensor values) or it tries to connect our module with the home-router with already typed in information. After successful configuration and connection the data as well as the configuration status (configured or not) is written in the EEPROM in case of power loss, so the user doesn't have to reconfigure every time the board had no power delivered. Therefore we had to implement a function that allows to write and read strings in the EEPROM, because you can only save and read byte by byte (values from 0-255). That is implemented as an int-function called "writeStringToEEPROM" and "readStringFromEEPROM". In the loop a timer starts to let the system know when to read values of the sensors, which is every ten minutes normally. After the first start up after power loss the sensor values are read one time directly. This happens in the "startMeasuring()"-function. After the values are read from the sensors they are directly send to an online database (thingspeak.com) over the active internet connection with our wifi module. To do that we use the "startSending()"-function. This repeats every few minutes so the user always has actual data that he can see from everywhere over the internet.

Our Code:

```
//----------------------------------------//
//SMARTPLANT - TEAM LF - EESTECH HACKATHON//
//----------------------------------------//



//-------------------------------------------------------------------------------------------//
//LIBRARIES//

#include <Arduino.h>
#include <SoftwareSerial.h>
#include <dht.h>
#include <EEPROM.h>
dht DHT;
//-------------------------------------------------------------------------------------------//




//-------------------------------------------------------------------------------------------//
//WIFI AND WEBSERVER//
const int RX = 0;
const int TX = 1;
String AP = "SmartPlant";       // AP NAME
String PASS = "SaveTheWorld"; // AP PASSWORD
String API = "GUP87Z8AK71MPWLR";   // Write API KEY
String HOST = "api.thingspeak.com";
String PORT = "80";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
SoftwareSerial esp8266(RX,TX);
```

```
//--------------------------------------------------------------------------------------------//
//
```

```
//--------------------------------------------------------------------------------------------//
//CAPACITIVE MOISTURE SENSOR//

const int moistureSensorPin = A1; // Analog pin sensor is connected to
const int dry = 600; //calibration Values for the sensor min/max
const int wet = 250; //calibration Values for the sensor min/max
int moistureSensorValue = 0; //somewhere to store the value read from the soil moisture sensor
int moisturePercentage = 0;

//--------------------------------------------------------------------------------------------//
```

```
//--------------------------------------------------------------------------------------------//
//TEMPERATURE AND HUMIDITY SENSOR//

const int tempHumiditySensorPin = 2; // digital pin sensor is connected to
int readTempHumiditySensor = 0; //somewhere to store the value read from the temperature and humidity sensor
float temperatureFloat = 0.0;
float humidityFloat = 0.0;
int temperatureInt = 0;
int humidityInt = 0;

//--------------------------------------------------------------------------------------------//
```

```
//-------------------------------------------------------------------------------------------//
//LIGHT SENSOR//

const int lightSensorPin = A0; // analog pin sensor is connected to
const int light = 250; //calibration Values for the sensor min/max
const int dark = 50; //calibration Values for the sensor min/max
int lightSensorValue = 0; //somewhere to store the value read from the soil moisture sensor
int lightPercentage = 0;
//-------------------------------------------------------------------------------------------//




//-------------------------------------------------------------------------------------------//
//NPN TRANSISTOR//

const int npnTransistor = 4; // digital pin transistor is connected to
//-------------------------------------------------------------------------------------------//




//-------------------------------------------------------------------------------------------//
//TIMER//

long checkIntervall = 600000; //after 10mins
long time;
unsigned long previousTime = 0;
//-------------------------------------------------------------------------------------------//
```

```
//--------------------------------------------------------------------------------------//
//GLOBAL VARIABLES//

bool startup = true;
bool configurationFinished;
int eepromOffset = 1;
char udp_packet[3];
//--------------------------------------------------------------------------------------//




//----------------------------------------VOID SETUP-----------------------------------------//
//--------------------------------------------------------------------------------------//
void setup() {
  Serial.begin(9600);

//--------------------------------------------------------------------------------------//
//WIFI AND WEBSERVER//
configurationFinished = EEPROM.read(0); //check if already was configurated at EEPROM address 0 (first byte)

if(configurationFinished){ //if was already configured before
  int newStr1AddrOffset = readStringFromEEPROM(eepromOffset, &AP); //read configured wifi data from EEPROM memory
  int newStr2AddrOffset = readStringFromEEPROM(newStr1AddrOffset, &PASS);
  int newStr3AddrOffset = readStringFromEEPROM(newStr2AddrOffset, &API);
  esp8266.begin(115200);
```

```
  sendCommand("AT",5,"OK");
  sendCommand("AT+CWMODE=1",5,"OK");  //set station-mode
  sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");
}else{ //if first time configuration
  esp8266.begin(115200);
  sendCommand("AT",5,"OK");
  sendCommand("AT+CWMODE=2",5,"OK");  //set access-point-mode
  sendCommand("AT+CWSAP=\""+ AP +"\",\""+ PASS +"\",\""+ 5 +"\",\""+ 3 +"\"",20,"OK"); //set SSID and PW for Access-Point
}

//-------------------------------------------------------------------------------------------//

} //end setup
//---------------------------------------END SETUP-----------------------------------------//
//-------------------------------------------------------------------------------------------//




//---------------------------------------VOID LOOP-----------------------------------------//
//-------------------------------------------------------------------------------------------//
void loop() {



if(configurationFinished == false){ //if was not already configured and now is acting as access point for receiving data
   if(esp8266.available()){
     for (int k = 0; k < 2; k++)
  {
  udp_packet[k] = esp8266.read(); //waiting for router SSID and Password data being submitted by user (via Access-Point connection)
  }
```

```
  AP = udp_packet[0];
  PASS = udp_packet[1];
  API = udp_packet[2];
  configurationFinished = true;
  }
}else{
  if(EEPROM.read(0) == false){ //if configuration wasn't already completed in the loops before
  int str1AddrOffset = writeStringToEEPROM(eepromOffset, AP); //write WIFI and API (Server) data to EEPROM if power is interrupted the data will be still available
  int str2AddrOffset = writeStringToEEPROM(str1AddrOffset, PASS);
  int str3AddrOffset = writeStringToEEPROM(str2AddrOffset, API);
  sendCommand("AT+CWMODE=1",5,"OK");  //set station-mode
  sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");
  EEPROM.write(0,1); //write first byte in EEPROM true to let the board know if it was alreasy configurated after power loss
  }
}

//--------------------------------------------------------------------------------------------//
//TIMER//

time = millis() - previousTime;

if(time >= checkIntervall) { //after time which is set (standard: 10min)
  previousTime = time;
  startMeasuring();
  startSending();
} else if(checkIntervall < 0) { //after time overflow
  previousTime = 0;
} else if(startup) { //if started up  measure sensor values one time directly
  startMeasuring();
  startSending();
```

```
}
//-------------------------------------------------------------------------------------------//


} //end loop
//----------------------------------------------END LOOP-----------------------------------------//
//-------------------------------------------------------------------------------------------//




//----------------------------------VOID STARTMEASURING-----------------------------------------//
//-------------------------------------------------------------------------------------------//
void startMeasuring() {

  digitalWrite(npnTransistor, HIGH); //turn on Transistor to power up sensors
  delay(10000); //delay to start up sensors




//-------------------------------------------------------------------------------------------//
//CAPACITIVE MOISTURE SENSOR//

moistureSensorValue = analogRead(moistureSensorPin);

moisturePercentage = map(moistureSensorValue, wet, dry, 100, 0);

if (moisturePercentage >= 100)
{
   moisturePercentage = 100;
}
```

```
else if (moisturePercentage <= 0)
{
  moisturePercentage = 0;
}
//--------------------------------------------------------------------------------------//




//--------------------------------------------------------------------------------------//
//TEMPERATURE AND HUMIDITY SENSOR//

readTempHumiditySensor = DHT.read11(tempHumiditySensorPin); //read DHT11 Sensor Signal
temperatureFloat = DHT.temperature;
humidityFloat = DHT.humidity;
temperatureInt = (int)temperatureFloat; //typecat float to int
humidityInt = (int)humidityFloat; //typecat float to int
//--------------------------------------------------------------------------------------//




//--------------------------------------------------------------------------------------//
//LIGHT SENSOR//

lightSensorValue = analogRead(lightSensorPin);

lightPercentage = map(lightSensorValue, dark, light, 0, 100);

if (lightPercentage >= 100)
{
   lightPercentage = 100;
```

```
}
else if (lightPercentage <= 0)
{
  lightPercentage = 0;
}
//-------------------------------------------------------------------------------------------//




digitalWrite(npnTransistor, LOW); //turn off Transistor to save energy
startup = false;

} //end measuring function
//-----------------------------------END STARTMEASURING-------------------------------------//
//-------------------------------------------------------------------------------------------//




//-----------------------------------VOID STARTSENDING--------------------------------------//
//-------------------------------------------------------------------------------------------//
void startSending(){
//WIFI MODULE//
 //used Data to transfer:
 // Moisture: moisturePercentage
 //Temperature: temperatureInt
 //Humidity: humidityInt
 // Light: lightPercentage
```

```
 String getData = "GET /update?api_key="+ API
+"&field1="+moisturePercentage+"&field2="+temperatureInt+"&field3="+humidityInt+"&field4="+li
ghtPercentage;
 sendCommand("AT+CIPMUX=1",5,"OK");
 sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");
 sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");
 esp8266.println(getData);delay(1500);countTrueCommand++;
 sendCommand("AT+CIPCLOSE=0",5,"OK");
}
//--------------------------------END STARTSENDING----------------------------------------//
//----------------------------------------------------------------------------------------//




//-----------------------------------VOID SENDCOMMAND-----------------------------------------//
//----------------------------------------------------------------------------------------//
void sendCommand(String command, int maxTime, char readReplay[]) {

  while(countTimeCommand < (maxTime*1))
  {
    esp8266.println(command);
    if(esp8266.find(readReplay))
    {
      found = true;
      break;
    }

    countTimeCommand++;
  }

  if(found == true)
```

```
  {
   Serial.println("OK");
   countTrueCommand++;
   countTimeCommand = 0;
  }

  if(found == false)
  {
   Serial.println("Fail");
   countTrueCommand = 0;
   countTimeCommand = 0;
  }

  found = false;
 } //end sendCommand function
 //------------------------------------END SENDCOMMAND-----------------------------------------//
 //-------------------------------------------------------------------------------------------//




//-----------------------------------INT WRITESTRINGTOEEPROM---------------------------------//
//-----------------------------------INT READSTRINGFROMEEPROM--------------------------------//
int writeStringToEEPROM(int addrOffset, const String &strToWrite)
{
  byte len = strToWrite.length();
  EEPROM.write(addrOffset, len);
  for (int i = 0; i < len; i++)
  {
    EEPROM.write(addrOffset + 1 + i, strToWrite[i]);
  }
  return addrOffset + 1 + len;
```

```
}


int readStringFromEEPROM(int addrOffset, String *strToRead)
{
int newStrLen = EEPROM.read(addrOffset);
char data[newStrLen + 1];
for (int i = 0; i < newStrLen; i++)
{
data[i] = EEPROM.read(addrOffset + 1 + i);
}
data[newStrLen] = '\0';
*strToRead = String(data);
return addrOffset + 1 + newStrLen;
}
//------------------------------------END WRITESTRINGTOEEPROM----------------------------------//
//------------------------------------END READSTRINGFROMEEPROM---------------------------------//
```