Republic of the Philippines
BOHOL ISLAND STATE UNIVERSITY
Calape Campus
San Isidro, Calape, Bohol

College Of Technology and Allied Sciences
INFORMATION AND COMMUNICATION TECHNOLOGY DEPARTMENT

Weather Monitoring with Rain Alarm

KETH DOMNIC TACATANI
JHUNDE DONASCO
BSCS-3A

Introduction:

Arduino is an open-source electronic prototyping platform that has revolutionized the world of microcontrollers. It provides a simple and accessible way for both beginners and experienced users to work with hardware and software, enabling the creation of interactive projects. The core component of Arduino is a programmable microcontroller board, which can be easily connected to various sensors, actuators, and other electronic components. With the help of the Arduino software, users can write and upload code to the board, instructing it on how to interact with the connected components.

In our project, "Weather Monitoring with Rain Alarm," we harness the power of Arduino to develop a system that continuously monitors weather conditions and alerts users when it detects rainfall. By utilizing different sensors, such as temperature, humidity, and raindrop sensors, we are able to collect real-time data on various weather parameters. This data is then processed by the Arduino board, allowing us to analyze and interpret the current weather conditions.

The versatility of Arduino allows us to customize the system according to our needs. We can program the Arduino board to perform specific actions based on the collected data. For example, when rainfall is detected, the system can trigger an alarm or send a notification to the user. Additionally, we can incorporate other features like data logging, wireless communication, or even integration with online weather services to enhance the functionality of our weather monitoring system.

Objective:

1. To design a Arduino based system to monitor the weather condition.
2. To provide an alert when it detects rainfall.
3. To measure the weather parameter such as temperature and humidity, heat index and wind speed..

Materials:

- **Arduino Board**- for running the code and control the other components.
- **Jumper Wires** - Connect various components on the breadboard using male-to-male and male-to-female jumper wires.
- **Breadboard or PCB** - Use a breadboard for prototyping or design a PCB (Printed Circuit Board) for a more permanent setup.

- **USB Cable-** Used to connect Arduino to the computer or a power source.
- **LDR(Light-Dependent Resistor)** - also known as photo resistor used to detect the presence or absence of rain to measure the intensity of ambient light.
- **Temperature and Humidity Sensor**- used to measure the ambient of temperature and humidity of environment. It's also the important piece of this project because it's provide the valuable data for weather monitoring.
- **Rain Drop Sensor** - used to detect the presence of a rain or detect a water droplets.
- **RTC(Real-Time Clock) –** used to track the current time and date.
- **Buzzer**- used to produced an alarm when a rainfall is detected.
- **10k ohm Resistors**- used to control the flow of electricity to the photoresistor.
- **1.8 TFT 160x128 Display –** used to display the reading from different sensors as well as the wind speed.

Procedure:

1. Prepare all the components and materials needed for the project. This includes the anemometer kit, rain drop sensor, photo resistor, temperature and humidity sensor, cylinder, photoresistor, Arduino board, LCD display, speaker, RTC (Real-Time Clock), cables, and a box for the base.

2. Begin building the anemometer. This involves attaching the blades or cups to a central hub or axle. Make sure the anemometer is securely assembled.

3. At the top of the anemometer, create a little box or platform to hold the additional sensors. This box should be sturdy and positioned in a way that it won't interfere with the rotation of the anemometer blades.

4. Attach the rain drop sensor and the photo resistor to the top of the box. The rain drop sensor will detect rain or moisture, while the photo resistor will measure the ambient light intensity.

5. Below the box, attach the temperature and humidity sensor. Ensure that it is properly sealed or positioned to prevent it from getting wet when it rains.

6. Create a cylinder that can rotate along with the anemometer. This cylinder should have an open area or slot on its side to allow the passage of light.

7. Inside the cylinder, attach a photoresistor. This photoresistor will detect the changes in light intensity as the cylinder rotates with the anemometer. This will detect the Rotation per Minute, that is used to calculate the wind speed.

8. Connect all the necessary cables correctly, using long cables if needed to reach different components. Pay attention to the pin configurations and ensure proper connections between the sensors, Arduino board, LCD display, speaker, and RTC.

9. Create another box that will serve as the base for the anemometer. This box will house the Arduino board, RTC, and other components. It should be designed to provide adequate space and protection for the electronics.

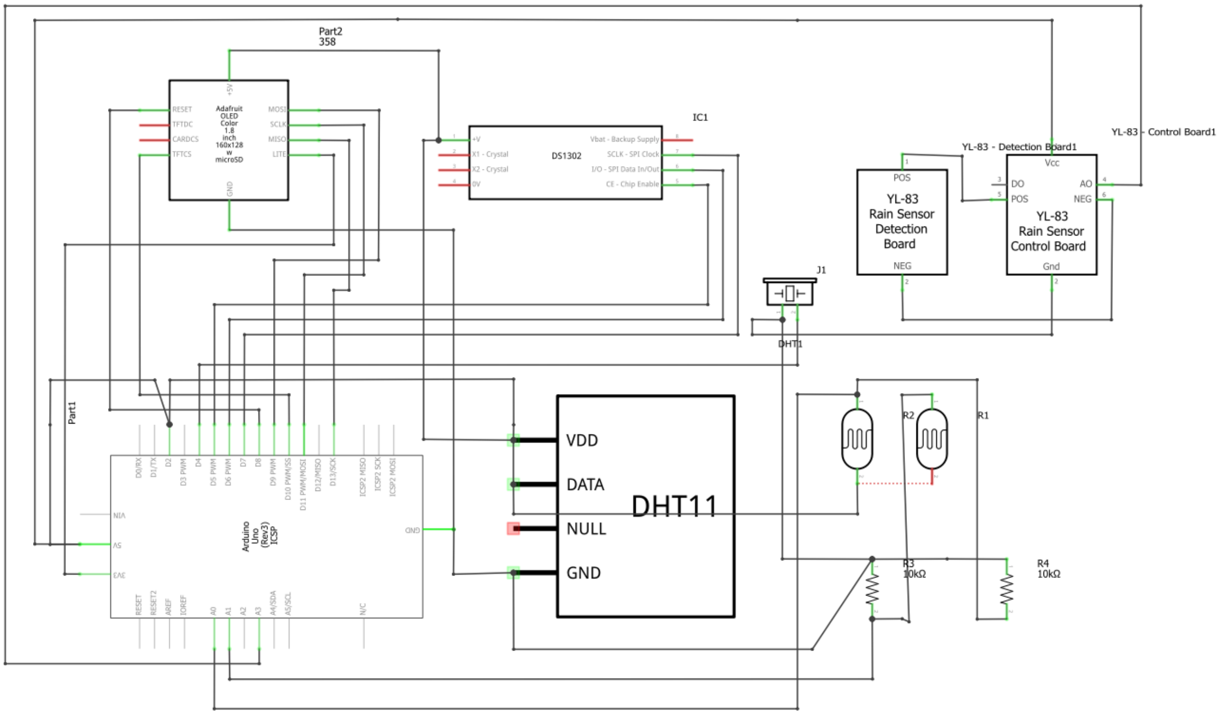
10. In the base box, create a square hole to accommodate the LCD display and a round hole for the speaker. Ensure that the holes are appropriately sized and positioned for easy visibility and sound output.

11. Attach the RTC (Real-Time Clock) inside the base box. The RTC will provide accurate time and date for the project.

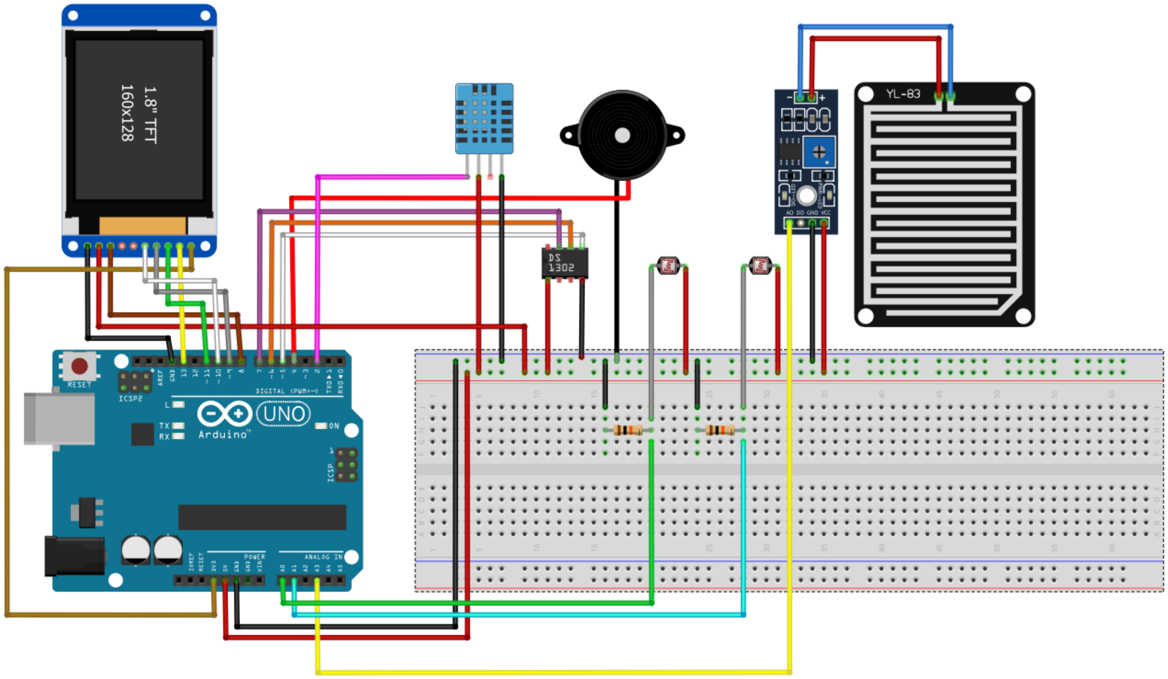
12. Connect all the cables from the sensors, TFT display, speaker, and RTC to the Arduino board. Double-check the connections to ensure they are correctly wired.

13. Upload the Arduino sketch or code that will control the operation of the anemometer and the sensors. This code will include instructions for reading sensor data, displaying information on the LCD, and producing sound from the speaker when necessary.

Schematic Diagram:



Circuit Diagram:



The Sketch:

```
  #include <virtuabotixRTC.h>
#include <TFT.h>
#include <SPI.h>
#include "DHT.h"
#include <math.h>

//for
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);


//for tft
#define cs   10
#define dc   9
#define rst  8

virtuabotixRTC myRTC(5, 6, 7);


//for TFT
TFT tft = TFT(cs, dc, rst);
char tftPrintout[20];
//for rtc
char minutePrintout[4];
char hourPrintout[4];
char weekDayPrintout[10];
char dayPrintout[4];
char monthPrintout[4];
char yearPrintout[6];
char amPmPrintout[4];
//for temp and hum
char temperaturePrintout[6];
char humidityPrintout[6];
char heatIndexPrintout[6];
//for raindrop
char rainDetectPrintout[15];
char rainDetectPrintout2[15];
//testPrintout
char testPrintout[70];

//for RTC time
String secondsTime;
String minutesTime;
String hourTime;
String weekDayTime;
String dayTime;
String monthTime;
String yearTime;
String amPmTime;



//for windspeed
const int maxSamples = 10;  //
Number of samples to average
unsigned long
rotationTimes[maxSamples];  //
Array to store rotation times
int sampleCount = 0;  // Counter
for the number of samples
collected
```

```
float rpm = 0;   // Variable to
store the time for one full
rotation
unsigned long lastRotationTime =
0;  // Variable to store the
time of the last rotation
const float decayRate = 0.65;
float windSpeed = 0;
const float radius =8.0;
unsigned long fullRotationTime =
0;

boolean detected = false;

//float rpm = 0;



//ldr rpm
int threshold=200;
int ldrRPM=A0;
int sensorValue=0;

//ldr clear sky
int ldrSky=A1;

//dht reads
int humidity;
int temperature;
int heatIndex;

//for raind drop
bool raining=false;
```

```
//buzzer
int buzzer = 4;
String playing= "";

//previousTime
unsigned long previousTime = -
60000;
unsigned long windPreviousTime =
-1100;
unsigned long windPreviousTime2
= 0;
unsigned long tempPreviousTime =
-11000;
unsigned long skyPreviousTime =
-11000;
unsigned long rainPreviousTime=
-11000;


void setup() {
Serial.begin(9600);   //
시리얼통신 초기화
pinMode(buzzer, OUTPUT);
tft.begin();
dht.begin();
tft.background(0,0,0);
tft.setRotation(2);
 tft.setTextSize(1);
  tft.stroke(255,255,255);
  tft.text("BISU Calape
Weather",2,2);
  tft.text("Temperature",3,65);
  tft.text("Sky",87,65);
  tft.text("Humidity",3,97);
  tft.text("Heat Index",67,97);
  tft.text("Wind Speed",3,128);
  tft.setTextSize(2);
//RTC.adjust(DateTime(2022,9,30,
18,59,00));  // 처음 한번만 적절한
날짜 시간으로 설정, 이후 주석처리
}

void loop() {



 //displayTFT();
  getTime();
    tft.setTextSize(2);
  checkWindSpeed();
  getTempHumidity();
  checkRain();
  getSkyState();
//delay(100);
//tft.fillScreen(0);
}




////////////////////////////////
/////////////////////////////
void checkWindSpeed(){


sensorValue=analogRead(ldrRPM);
//    Serial.print("Sensor value
="); //Show “Sensor value=” on
serial monitorde
// Serial.println(sensorValue);

  if(sensorValue>threshold &&
detected==false){
     detected=true;

      Serial.print("Sensor value
="); //Show “Sensor value=” on
serial monitorde
 Serial.println(sensorValue);
    unsigned long currentTime =
micros();  // Get the current
time in microseconds
    rotationTimes[sampleCount %
maxSamples] = currentTime;  //
Store the rotation time in the
array

    sampleCount++;  // Increment
the sample count

    // Calculate the number of
samples to use for averaging
    int numSamples =
min(sampleCount, maxSamples);

    // Calculate the total
rotation time
    unsigned long
totalRotationTime =
rotationTimes[(sampleCount - 1)
% maxSamples] -
rotationTimes[(sampleCount -
numSamples) % maxSamples];

    // Calculate the average
rotation time and convert it to
RPM
    float averageRotationTime =
(float)totalRotationTime /
numSamples;
     rpm = 56858000.0 /
averageRotationTime;


    lastRotationTime = millis();

    windSpeed = ((2 * 3.14159 *
radius * rpm * 60) / 100000)*6;
    if (isnan(windSpeed) ||
isinf(windSpeed)) {
    windSpeed = 0.0; // Set the
value to 0 if it is NaN (Not a
Number) or infinity
    }
//  Serial.println("rpm is
"+String(rpm));
//    Serial.println("Wind
Speed:"+String(windSpeed)+"km/hr
");
    String
windSpeedStr=String(windSpeed)+"
km/hr";

windSpeedStr.toCharArray(tftPrin
tout,20);
    if(millis() -
windPreviousTime <1000){
    return;
  }
  windPreviousTime=millis();
     tft.fillRect(10,140, 117,
15, 0);

tft.text(tftPrintout,10,140);


  }
  else if
(sensorValue<threshold){
    detected=false;
  }


  if(millis() -
windPreviousTime2 <1000){
    return;
  }
  windPreviousTime2=millis();

  if (millis() -
lastRotationTime > 5000 && rpm >
1) {
    rpm *= decayRate;  // Apply
the decay rate to gradually
reduce the RPM
```

```
    windSpeed = ((2 * 3.14159 * radius * rpm * 60) / 100000)*6;
     if (isnan(windSpeed) || isinf(windSpeed)) {
      windSpeed = 0.0; // Set the value to 0 if it is NaN (Not a Number) or infinity
    }
//  Serial.println("rpm is "+String(rpm));
//    Serial.println("Wind Spd:"+String(windSpeed)+"km/hr");

  String windSpeedStr=String(windSpeed)+" km/hr";



windSpeedStr.toCharArray(tftPrintout,20);

//
//String test=String(windPreviousTime2)+" prev";
//test.toCharArray(testPrintout,20);
//  tft.fillRect(70,128, 50, 9, 0);
//  tft.setTextSize(1);
// tft.text(testPrintout,70,128);
//tft.setTextSize(2);


    tft.fillRect(10,140, 117, 15, 0);

tft.text(tftPrintout,10,140);



  }


}

///////////////////////////////////////////////////////////////
void getTime(){

  if(millis() - previousTime <58000){
    return;
  }
  previousTime=millis();
  tft.fillRect(26, 19, 77, 41, 0);
 myRTC.updateTime();

  // Convert hours to 12-hour format
  int hours12 = myRTC.hours;
  bool isPM = false;
  if (hours12 >= 12) {
```

```
  isPM = true;
  if (hours12 > 12) {
   hours12 -= 12;
  }
 }
 if (hours12 == 0) {
  hours12 = 12;
 }



 int dayofweek=myRTC.dayofweek;
 if(dayofweek==1){
  weekDayTime="Sunday";
 }else if(dayofweek==2){
  weekDayTime="Monday";
 }else if(dayofweek==3){
  weekDayTime="Tuesday";
 }else if(dayofweek==4){
  weekDayTime="Wednesday";
 }else if(dayofweek==5){
  weekDayTime="Thursday";
 }else if(dayofweek==6){
  weekDayTime="Friday";
 }else if(dayofweek==7){
  weekDayTime="Saturday";
 }

int monthTimeInt=myRTC.month;
 if(monthTimeInt==1){
  monthTime="Jan";
 }else if(monthTimeInt==2){
  monthTime="Feb";
 }else if(monthTimeInt==3){
```

```
  monthTime="Mar";
 }else if(monthTimeInt==4){
  monthTime="Apr";
 }else if(monthTimeInt==5){
  monthTime="May";
 }else if(monthTimeInt==6){
  monthTime="Jun";
 }else if(monthTimeInt==7){
  monthTime="Jul";
 }else if(monthTimeInt==8){
  monthTime="Aug";
 }else if(monthTimeInt==9){
  monthTime="Sep";
 }else if(monthTimeInt==10){
  monthTime="Oct";
 }else if(monthTimeInt==11){
  monthTime="Nov";
 }else if(monthTimeInt==12){
  monthTime="Dec";
 }


//   secondsTime
=String(myRTC.seconds);
 minutesTime = myRTC.minutes <
10?
"0"+String(myRTC.minutes):String
(myRTC.minutes);
 hourTime =hours12 < 10?
"0"+String(hours12):hours12;
 dayTime =myRTC.dayofmonth <
10?
```

```
"0"+String(myRTC.dayofmonth):Str
ing(myRTC.dayofmonth);
  yearTime =String(myRTC.year);
  amPmTime=isPM?"PM":"AM";
  tft.setTextSize(2);



hourTime.toCharArray(hourPrintou
t,4);

tft.text(hourPrintout,30,22);

    tft.text(":",58,24);


minutesTime.toCharArray(minutePr
intout,4);

tft.text(minutePrintout,67,22);

    tft.setTextSize(1);

amPmTime.toCharArray(amPmPrintou
t,4);

tft.text(amPmPrintout,91,29);


monthTime.toCharArray(monthPrint
out,4);

tft.text(monthPrintout,31,40);
```

```

dayTime.toCharArray(dayPrintout,
4);
    tft.text(dayPrintout,53,40);

    tft.text(",",63,41);


yearTime.toCharArray(yearPrintou
t,6);

tft.text(yearPrintout,75,40);

    int textWidth =
weekDayTime.length() * 6;
     int textX = (tft.width() -
textWidth) / 2;

weekDayTime.toCharArray(weekDayP
rintout,10);

tft.text(weekDayPrintout,textX,5
0);
}
////////////////////////////////
////////////////
void getTempHumidity()
{

  if(millis() - tempPreviousTime
<10000){
    return;
  }
  tempPreviousTime=millis();
//  if(isnan(humidity) ||
isnan(temperature) {
//    Serial.println("Failed to
read from DHT sensor!");
//    return;
//  }

  temperature =
int(dht.readTemperature());
  humidity =
int(dht.readHumidity());
   float h = dht.readHumidity();
  float t =
dht.readTemperature();
   float hic =
dht.computeHeatIndex(t, h,
false);
  heatIndex = round(hic);

  //clear
   tft.fillRect(13,78, 50, 16,
0);
   tft.fillRect(13,109, 50, 16,
0);
   tft.fillRect(77,109, 50, 16,
0);

  String temp=
String(temperature)+"\367C";

temp.toCharArray(temperaturePrin
tout,6);

tft.text(temperaturePrintout,13,
78);

String hum=
String(humidity)+"%";

hum.toCharArray(humidityPrintout
,6);

tft.text(humidityPrintout,13,109
);

String heat=
String(heatIndex)+"\367C";

heat.toCharArray(heatIndexPrinto
ut,6);

tft.text(heatIndexPrintout,77,10
9);
}
////////////////////////////////
////////////////
void checkRain(){

  if(millis() - rainPreviousTime
<5000){
    return;
  }
  rainPreviousTime=millis();

    int value =
analogRead(A3);//read value
   // Serial.println(value);
    String rainEq="";
    String rainEq1="Rain";

    bool isSound=false;
     tft.setTextSize(1);
     if (value < 1000){
      raining = true;
       tft.fillRect(64,75, 65,
19, 0);
     }
     else{
      raining = false;
      playing = "";
      return;
     }
     if(windSpeed > 25){
      rainEq="Light";
        soundBuzzer("Light",2);
        playing="Light";
     }

    else if (value < 1000 &&
value >500) {//check condition
      rainEq="Light";
        soundBuzzer("Light",2);
        playing="Light";
```

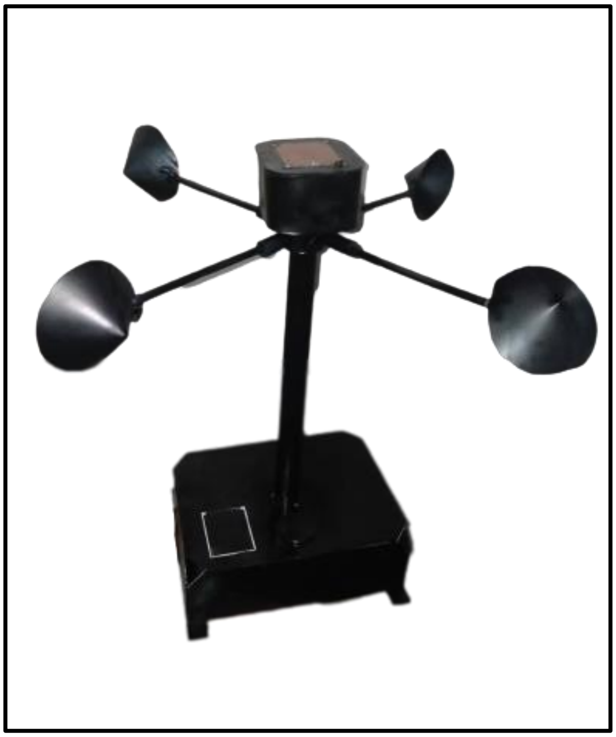
```
    }
    else if (value <= 500 &&
value > 300) {//check condition
      rainEq="Moderate";
      soundBuzzer("Moderate",3);
        playing="Moderate";
    }
    else if (value <= 300)
{//check condition
      rainEq="Heavy";
      soundBuzzer("Heavy",4);
        playing="Heavy";
    }

      int textWidth =
rainEq.length() * 6;
      int startingX = (122 + 67)
/ 2 - textWidth / 2;

rainEq.toCharArray(rainDetectPri
ntout,15);

tft.text(rainDetectPrintout,star
tingX,75);

      textWidth =
rainEq1.length() * 6;
      startingX = (122 + 67) / 2
- textWidth / 2;

rainEq1.toCharArray(rainDetectPr
intout2,15);
```

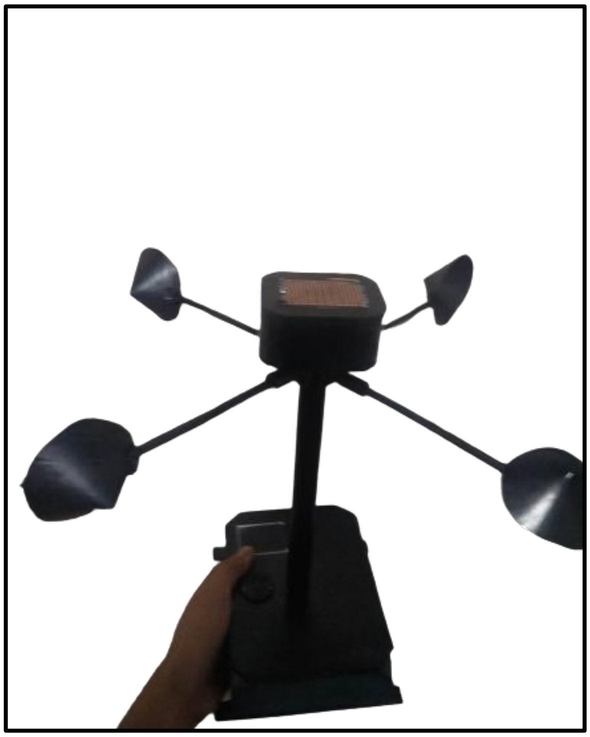
```
tft.text(rainDetectPrintout2,sta
rtingX,83);



//    else{
//      int textWidth =
rainEq.length() * 6;
//      int startingX = (122 +
67) / 2 - textWidth / 2;
//
rainEq.toCharArray(rainDetectPri
ntout,15);
//
tft.text(rainDetectPrintout,star
tingX,80);
//    }
}

void soundBuzzer(String
playing2, int count){
      if(playing2!=playing){
        for(int
i=0;i<count;i++){
        tone(buzzer, 400);  //
Start the buzzer at a frequency
of 1000 Hz
        delay(400);  // Keep the
buzzer on for 0.5 seconds
        noTone(buzzer);  // Stop
the buzzer
        delay(100);  // Delay
for 0.5 seconds
```

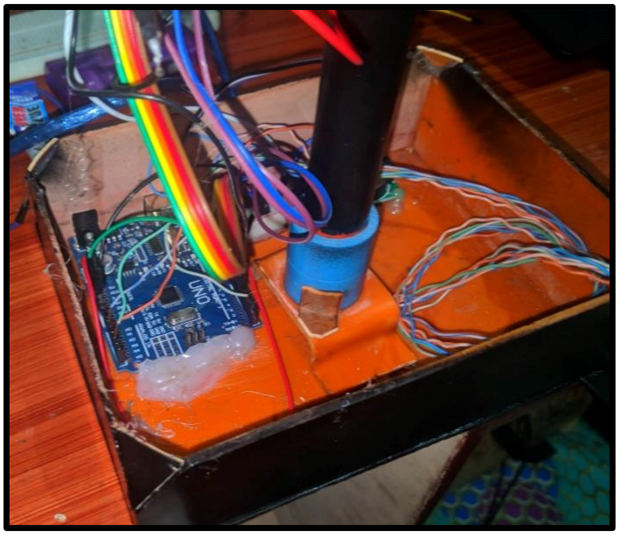
```
        }
      }
}
////////////////////////////////
///////////////////
void getSkyState(){
  tft.setTextSize(1);
  if(millis() - skyPreviousTime
<10000){
    return;
  }
   tft.fillRect(64,75, 50,
19,0);
  skyPreviousTime=millis();

  if(raining){
    return;
  }
  String sky;
  int
skyValue=analogRead(ldrSky);
//  Serial.print("Sky ");
//  Serial.println(skyValue);

  if(myRTC.hours<8 ||
myRTC.hours>16){
    sensorValue *=1.20;
  }

  if(skyValue>=700 &&
skyValue<1200){
    sky="Sunny";
  }
  else if(skyValue>= 500 &&
skyValue <700){
    sky="Clear";
  }
  else if(skyValue>= 100 &&
skyValue <500){
    sky="Cloudy";
  }
  else if(skyValue<100){
    sky="Dark";
  }
}
```

```
    int textWidth = sky.length()
* 6;
      int startingX = (127 + 67)
/ 2 - textWidth / 2;

sky.toCharArray(rainDetectPrinto
ut,15);

tft.text(rainDetectPrintout,star
tingX,80);
```

The Project:







Analysis, Conclusion, and Recommendation:

The project "Weather Monitoring with Rain Alarm" is not only fun and interesting but also highly educational, as it provides an opportunity to expand our knowledge and skills in the fields of electronics and programming. By working on this project, we are exposed to new components such as the raindrop sensor and temperature and humidity sensor, which may be unfamiliar to us initially. This allows us to delve into the functionalities and applications of these sensors, broadening our understanding of their importance in weather monitoring.

The temperature and humidity sensor integrated into the system provides valuable information for understanding the environment and making informed decisions. Additionally, the collected data can be used for scientific analysis, climate research, and even for personal interest in studying local weather patterns.

In conclusion, the "Weather Monitoring with Rain Alarm" project offers an engaging and practical way to learn about electronics, programming, and the significance of weather monitoring. By combining various sensors and an Arduino board, we can create a functional system that provides real-time weather data, alerts, and valuable insights. This project has the potential to make a positive impact in agriculture, outdoor activities, and scientific endeavors related to weather analysis.

For further improvement and enhancement of the project, consider exploring additional features or functionalities. For example, integrating a wind direction sensor could provide a more comprehensive weather monitoring system. Incorporating wireless communication capabilities would enable remote access and control of the system, allowing users to monitor weather conditions from a distance. Furthermore, implementing a data logging feature could facilitate long-term data analysis and trend identification.

Curriculum Vitae:

First name: Keth Dominic

Last name: Tacatani

Middle name: Baquial

Gender: Male

Address: Purok 5, Bentig, Calape, Bohol

EDUCATIONAL BACKGROUND:

Primary Level

Bentig-Calunasan, Elementary school

Calunasan, Calape, Bohol

S.Y. 2013-2014

Junior High School

Mayor Anunciacion R. Tuazon National School of Fisheries

Calunasan, Calape, Bohol

S.Y. 2014-2018

Senior High School

Mayor Anunciacion R. Tuazon National School of fisheries

Calunasan, Calape, Bohol

S.Y. 2019-2020

Name: Jhunde B. Donasco

Gender: Male

Address: Catmonan, Calape, Bohol

Educational Background

Primary Level

Bon-bon Elementary
School

2013-2014

Bonbon, Calape, Bohol

Junior High School

Calape National High School

2017-2018

Sta. Cruz, Calape, Bohol

Senior High School

Calape National High School

2019-2020