Project no: 01	DESIGN AND DEVOLOP THE HOME AUTOMATION BASED APPLICATION
Date:	Rainfall Detector

OBJECTIVE

- To develop an IoT-based rainfall detection system using ultrasonic sensors and ESP32 microcontrollers for real-time precipitation monitoring via a cloud platform.
- To develop the product that helps in development of 13th SDG(Climate action).

AIM

To create an IoT-based rainfall measurement system using ultrasonic sensors and ESP32 microcontrollers for real-time monitoring. It seeks to improve decision-making and resilience against weather-related risks by providing timely and accuraterainfall data through a cloud-based platform.

INTRODUCTION

In response to the pressing need for effective rainfall monitoring systems, this project endeavors to introduce an IoT-based solution for accurate and timely measurement of precipitation levels. Heavy rainfall, a prevalent weather hazard, poses significant risks to communities, infrastructure, and the environment, necessitating proactive measures for disaster mitigation. Traditional methods of rainfall detection often suffer from limitations such as inadequate coverage, delayed response times, and inefficient resource allocation. Leveraging IoT technology, this project aims to overcome these challenges by deploying a network of ultrasonic sensors and ESP32 microcontrollers for real-time data collection and analysis. By transmitting data to a cloud-based platform, authorities can promptly assess rainfall intensity and implement precautionary measures to mitigate potential risks. Ultimately, this project seeks to enhance resilience against weather-related hazards by providing accurate and actionable rainfall data for informed decision-making in various sectors.

Block Diagram

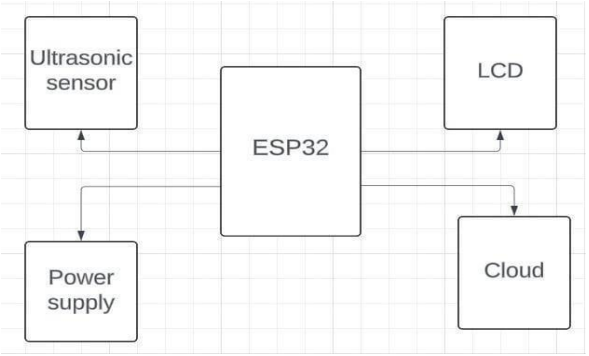

Figure 1.4: Block Diagram For Rainfall Detector

Circuit Diagram

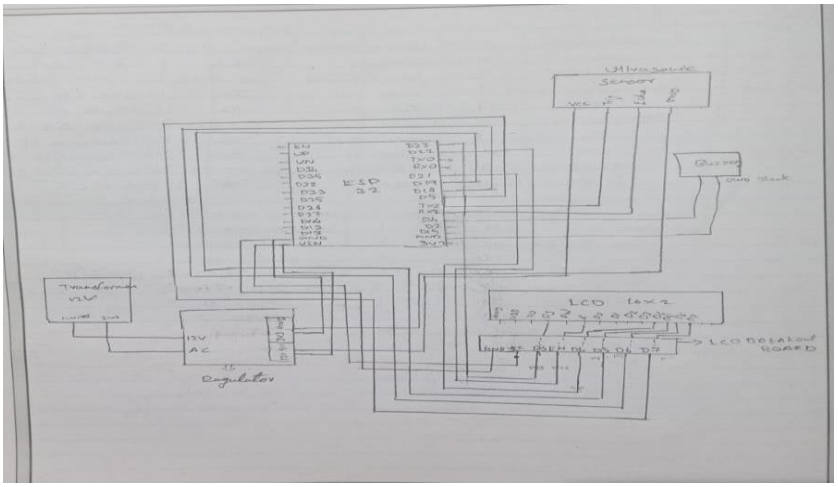

Figure 1.5: Circuit Diagram For Rainfall Detector

SUPPORTING SYSTEM – SOFTWARE DETAILS/HARDWARE DETAILS
SOFTWARE DETAILS

Arduino IDE for ESP32 Program

Arduino Integrated Development Environment (IDE) is an open source IDE that allows users to write code and upload it to any Arduino board. Arduino IDE is written in Java and is compatible with Windows, macOS and Linux operating systems.

Google Cloud IoT platform

A cloud platform enables businesses to rent access to computing resources on demand over the internet with pay-as-you-go pricing, rather than buying, installing, and managing their own data centers, servers, and software required to have these resources available on premises.

HARDWARE DETAILS

Ultrasonic Sensor

Figure 1.1: Ultrasonic Sensor

The An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect across boundaries to produce distinct echo patterns.

ESP32

Figure 1.2: Esp32 Development kit

The ESP32 is a powerful microcontroller chip designed by Espresso Systems, offering a versatile solution for a wide range of applications. It features a dual-core Ten silica LX6 microprocessor, running at up to 240 MHz, which enables efficient multitasking and real-time processing capabilities. With integrated Wi-Fi and Bluetooth connectivity, including support for BLE (Bluetooth Low Energy), the ESP32 facilitates seamless communication with other

devices and networks, making it ideal for IoT (Internet of Things) projects and connected applications.

LCD DISPLAY

Figure 1.3: LCD Display

An LCD (Liquid Crystal Display) shown in figure 3.4 is a16x2 consists of 16 columns and 2 rows of characters, typically alphanumeric. It works by applying electrical signals to control liquid crystals, causing them to change orientation and selectively block or allow light to pass through, forming characters or graphics on the display. Each character position is addressed individually, allowing for the display of text, numbers, symbols, and basic graphics. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for the playing custom characters, special and even animations, etc.

PROGRAM

```
//#include "HX711.h"
#include
<LiquidCrystal.h>
LiquidCrystal
lcd(23,22,21,19,18,5);
#include<WiFi.h>
#include
<WiFiClient.h>
#include
<HTTPClient.h>
#define                    u1_triger_high
digitalWrite(17,HIGH)          #define
u1_triger_low      digitalWrite(17,LOW)
#define ultra1 16
#define alm 4
int
ult,u2,ss=0,sec=0,secc=
```

```
0;WiFiClient client;
HTTPClient http;
//#define LED 6
const char *ssid =
"kamalesh"; const char *pass
=  "kamalesh07";
const char* server = "mangocity.appblocky.com";
const char* serverName = "mangocity.appblocky.com";
void Lcd_Decimal3(unsigned char com, unsigned char com1, unsigned int val){
.setCursor(com,
com1);lcd.print(val);
}
//int
sec=0,secc=0,u1;
String  data1 =
""; void setup()
{
Serial.begin(115
200);delay(10);
// myservo.attach(4);
Serial.println("Connecting
to ");Serial.println(ssid);
WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED)
{
delay(1000
);
Serial.print
(".");
}
Serial.println("");
Serial.println("WiFi
connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
Serial.print("Netmask: ");
Serial.println(WiFi.subnetMa
sk());Serial.print("Gateway:
");
Serial.println(WiFi.gatewayIP
());
Serial.println("cONNECT111
1");
//sensors.begin();
pinMode(alm,OUTP
```

```
UT);
digitalWrite(alm,LO
W);
//     digitalWrite(rly2,HIGH);
//     pinMode(rly3,OUTPUT);
//     digitalWrite(rly3,HIGH);
//     pinMode(rly3,OUTPUT);
//     pinMode(rly4,OUTPUT);
//
 digitalWrite(rly4,HIGH);
pinMode(17,OUTPUT);
pinMode(16,INPUT);
lcd.begin(16, 2);
lcd.setCursor(0,0);
lcd.print(".............");
lcd.setCursor(0,1);
lcd.print("--------- "); delay(2000);
lcd.clear();}

void loop()
{

ult=read_ultrasoni
c1();
lcd.setCursor(0,0)
;
lcd.print("LEVEL
:");
// Lcd_Decimal3(2,0,ult);

u2=10-ult;
if(ult>=10)u2=0;
Lcd_Decimal3(6,0
,u2);
lcd.setCursor(9,0);
lcd.print(" Cm");
delay(20);
if(u2>5)
{ ss=1;
lcd.setCursor(0,
1);
lcd.print(" RAINFALL_HIGH
");digitalWrite(alm,HIGH);
data1="RAINFALL_HIGH";
http_send1();
```

```
delay(1000);
digitalWrite(alm,HI
GH);
}
lcd.setCursor(0, 1);
digitalWrite(alm);
lcd.print("NORML
");
data1="NORMAL"
;
}
sec++;
//if(sec>1){secc++;sec=0;se1++;}
if(sec>2){http_send1();delay(500);sec=0;sec
c=0;}
}
unsigned int read_ultrasonic1()
{
int ultrasonic=0;
u1_triger_low;dela
y(1);
u1_triger_high;
delay(5);
u1_triger_low;
ultrasonic=pulseIn(ultra1,HIGH)/56.0;
 ultrasonic;
}
void http_send1()
{
if (WiFi.status() ==
WL_CONNECTED) {WiFiClient
client;
HTTPClient http;
http.begin(client,"http://mangocity.appblocky.com/webdb/storeavalue.php?tag=nsh0252&valu
e=" +String(u2) + ","  + String(data1));
int httpCode = http.GET();  //Send the request
String data = http.getString();        //Get the
response dataSerial.println(data);  //Print request
response data http.end(); delay(100);
}
```

OUTPUT

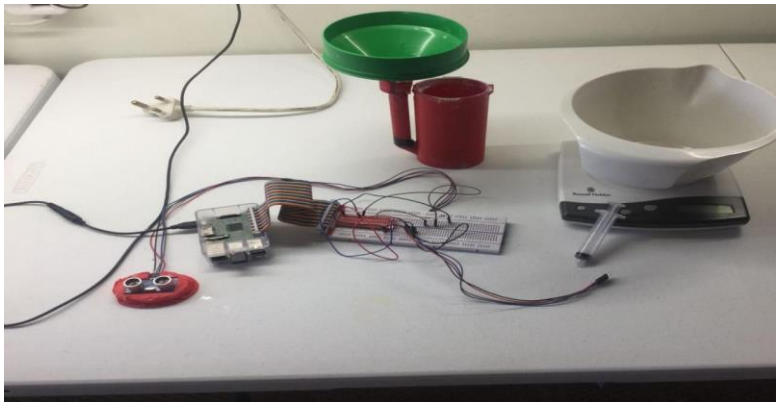

Figure 1.6: Output of Rainfall Detector

RESULT

The developed IoT-based rainfall measurement system effectively detected precipitation levelsin real-time. Data analysis showcased the system's accuracy in assessing rainfall intensity, enabling timely precautionary measures. Overall, the project demonstrated the system's potential to enhance resilience against weather-related hazards.

CONCLUSION AND FUTURE SCOPE

In conclusion, the IoT-based rainfall measurement system presented a viable solution for real-time monitoring of precipitation levels. Its implementation showcased improved decision-making capabilities and enhanced resilience against weather-related risks. Movingforward, further refinement and widespread adoption of such systems could significantly bolster disaster mitigation efforts.