Semester 6

2CSDE66 - Internet Of Things

Innovative Assignment

Project Title:

Smart Water Tank

Submitted by: - **Gunjan Vinzuda (18BCE257)**

Chandni Tomar (18BCE246)

I. Hardwares Used:

II. ESP-32 Dev Module
III. Ultrasonic Sensor HC-SR04
IV. 3 LEDs
V. Jumper Wires

II. Code:

```
#include "ThingSpeak.h"

#include <WiFi.h>


char ssid[] = "D-Link"; //SSID

char pass[] = "01234568"; // Password


// --------------- Tank details --------------//

const int total_height = 15.6; // Tank height in CM

const int hold_height = 13.5;// Water hold height in CM

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


//----- minutes -----//

float minute = 0.5; // Data update in min.

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


//----------- Channel Details -------------//

unsigned long Channel_ID = 1358838; // Channel ID

const int Field_number = 1; // To which field to write data

const char * WriteAPIKey = "51OYSWFLR8E224DH"; // Write API Key

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

```
const int trigger = 2;

const int echo = 5;


int led1 = 25;

int led2 = 26;

int led3 = 27;




long Time;

int x;

int i;

float distanceCM;

float resultCM;

int tnk_lvl = 0;

float sensr_to_wtr = 0.00;

WiFiClient  client;




void setup()

{

  Serial.begin(115200);

  pinMode(trigger, OUTPUT);

  pinMode(echo, INPUT);

  WiFi.mode(WIFI_STA);

  ThingSpeak.begin(client);

  sensr_to_wtr = total_height - hold_height;
```

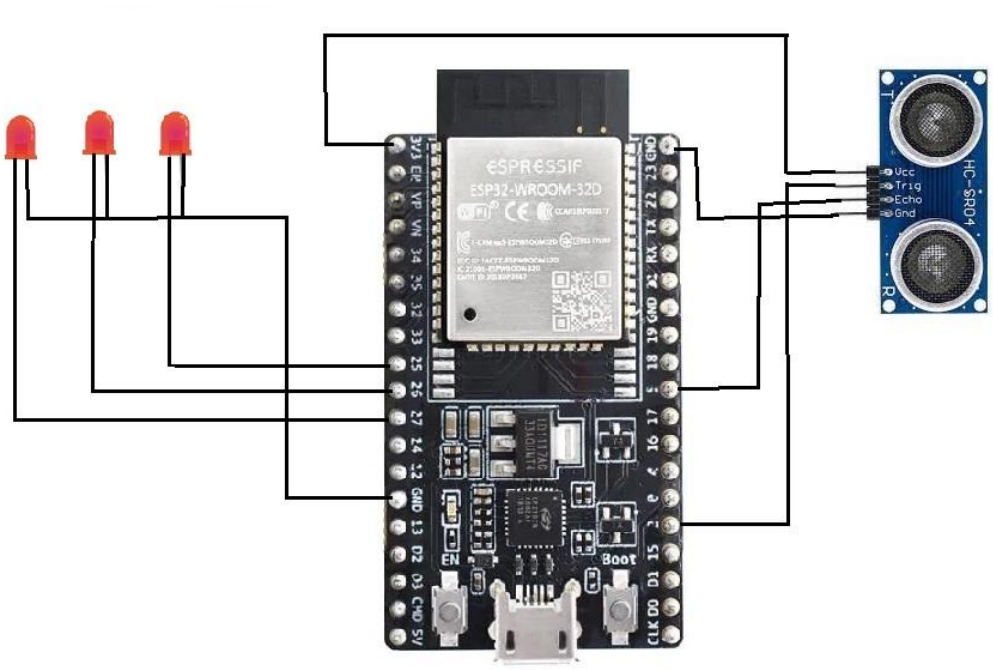
```
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
 internet();
 delay(1000);
}
void loop()
{
    Serial.println("System Standby....");
    Serial.print(i);
    Serial.println(" Minutes elapsed.");
    delay(1000);

  measure();
  Serial.println(Time);
  Serial.print("Tank Level:");
  Serial.print(tnk_lvl);
  Serial.println("%");
  upload();
}
void upload()
{
  internet();
```

```
  x = ThingSpeak.writeField(Channel_ID, Field_number, tnk_lvl,
WriteAPIKey);
  if (x == 200)Serial.println("Data Updated.");
  if (x != 200)
  {
    Serial.println("Data upload failed, retrying....");
    delay(15000);
    upload();
  }
}


void measure()
{
  delay(100);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);
  Time = pulseIn(echo, HIGH);
  distanceCM = Time * 0.034;
  resultCM = distanceCM / 2;


  tnk_lvl = map(resultCM, sensr_to_wtr, total_height, 100, 0);


  if (tnk_lvl > 100) tnk_lvl = 100;
  if (tnk_lvl < 0) tnk_lvl = 0;


  if(  (tnk_lvl > 66) && (tnk_lvl <= 100)   )
```

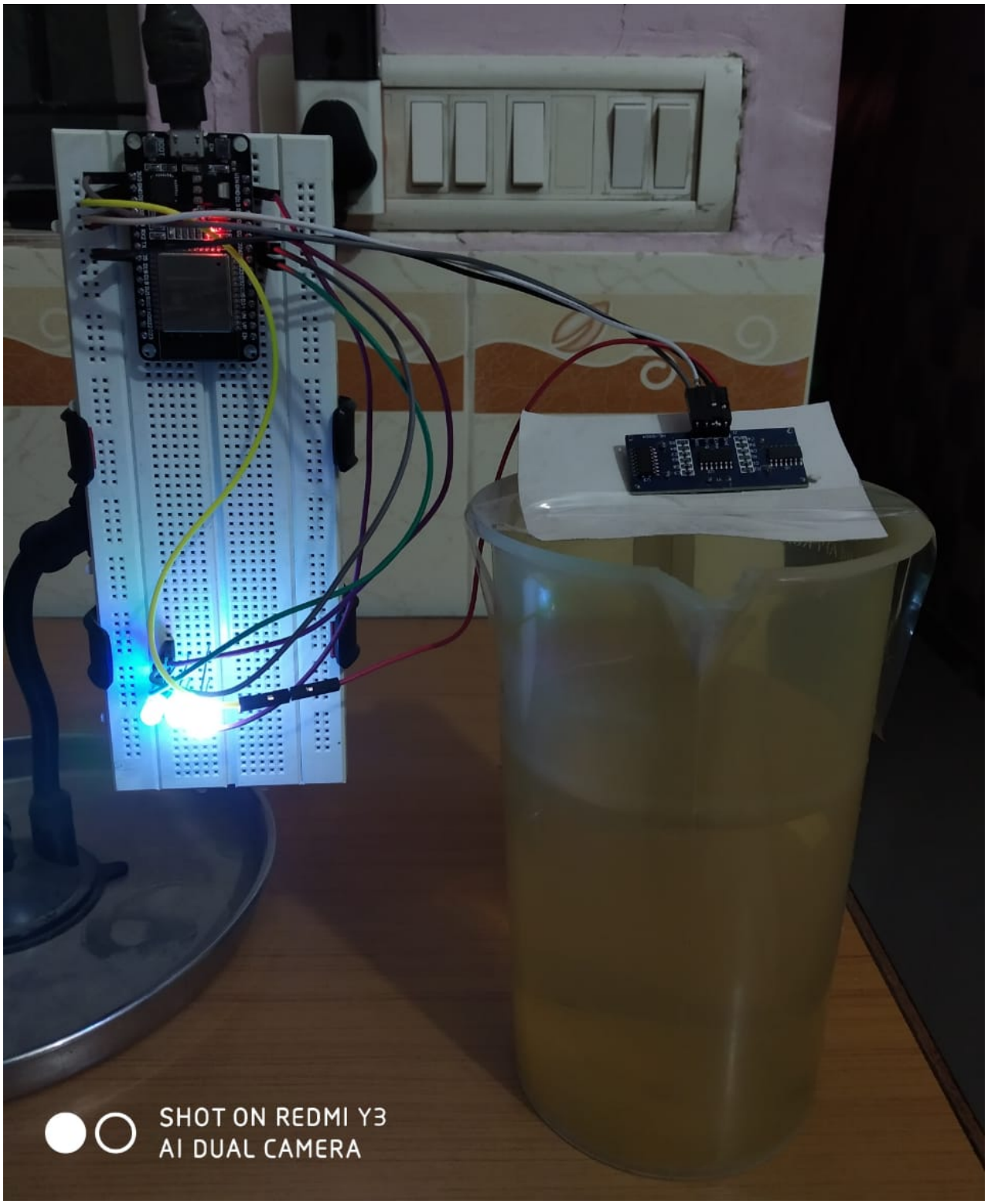
```
{
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
} else
if(  (tnk_lvl > 33) && (tnk_lvl <= 66)  )
{

  digitalWrite(led1, LOW);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
 } else

if(  (tnk_lvl > 0) && (tnk_lvl <= 33)  )
{

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, HIGH);
 }
 else

if(  tnk_lvl == 0 )
{

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
```

```
  digitalWrite(led3, LOW);
 }
}


void internet()
{
  if (WiFi.status() != WL_CONNECTED)
  {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    while (WiFi.status() != WL_CONNECTED)
    {
      WiFi.begin(ssid, pass);
      Serial.print(".");
      delay(5000);
    }
    Serial.println("\nConnected.");
  }
}
```

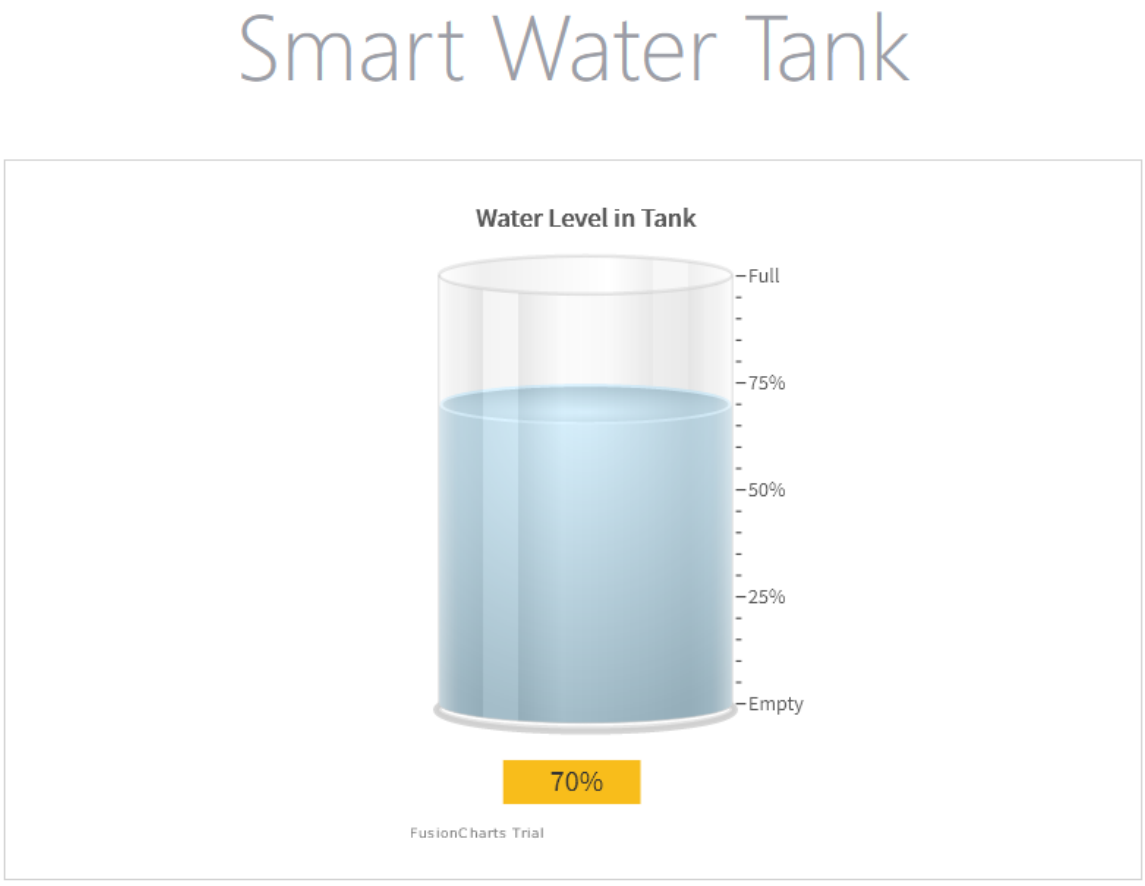
III. Circuit Diagram



IV. Project Set-up



VI. Water Tank Simulation



VII. Html Code:

```
<html>
<head>
    <title>Smart Water Tank</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <script src="https://smtpjs.com/v3/smtp.js"></script>
    <script type="text/javascript" src="https://cdn.fusioncharts.com/fusioncharts/latest/fusioncharts.js"></
    script>
    <script type="text/javascript" src="https://cdn.fusioncharts.com/fusioncharts/latest/themes/
    fusioncharts.theme.fusion.js"></script>

    <link rel="stylesheet" type="text/css" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/
    bootstrap.min.css">
    <link rel="stylesheet" type="text/css" href="./main.css">
    </head>
    <body>
        <div class="container">
            <h1 class="m-5">Smart Water Tank</h1>
            <div class="box_1" id="chart-container" align="center">
                WaterTank will load here!
            </div>
            <p class="artical-content mt-5" >This project is preprared by <strong>Chandni Tomar (18BCE246)</
            strong> and <strong>Gunjan Vinzuda (18BCE257)</strong> as a part of Innovative Assignment for <em>
            Internet of Things</em> subject.<br>
            In this project, we are simulating the idea of a smart water tank. Using this idea we can check the
            water level of our house water tank, anytime, anywhere!!<br>
            In this project, the remote sensor data is uploaded to the cloud. Here we are using ThingSpeak as
            cloud storage. The data is uploaded using ESP32. The value is fetched by web page and according to
            the value, the simulation of the water tank changes the water level. Also when the water level of
            the tank is below from certain value it will notify you with an email.<br>
```

```
            The ultrasonic distance sensor is placed on the tank, and it is connected to ESP32. The sensor
            takes the reading in the time interval of 1 second and ESP32 uploads the value on ThingSpeak cloud.
            For the simulation, JavaScript is used to take the last reading from the cloud and update the water
            level in the simulation.</p> <br>
            <h3>Circuit Diagram</h3>
            <p>
                The pin diagram of the circuit is illustrated as below. <br>
            </p>
            <img src="circuit digram.jpg">
            <p>
                As it is shown in the diagram the trigger and echo are connected to pins 2 and 5 of ESP32
                respectively. The VCC pin of the sensor is connected to the 3V3 pin of ESP32. The LEDs are
                connected to pins 25, 26 and 27 of ESP32. <br> When the water level is between 66% to 100% all
                the LEDs will be on. When the water level is between 33% to 66% the LEDs connected to pins 26
                and 27 will be turned on. When the water level is below 33% only the LED connected to pin 27
                will be on. When the water level is 0, i.e. tank is empty all the LEDs will be off. Also when
                the water level is below 33% the notification is will be sent through email.
            </p>
        </div>
    <script type="text/javascript" src="water_tank.js"></script>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/
    X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="
    sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="
    sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
    </body>
</html>
```

VIII. JavaScript

```
var waterLevel;
var flag=true;

FusionCharts.ready(function(){
            var chartObj = new FusionCharts({
    type: 'cylinder',
    dataFormat: 'json',
    renderAt: 'chart-container',
    width: '300',
    height: '450',
    dataSource: {
        "chart": {
            "theme": "fusion",
            "caption": "Water Level in Tank",
            "subcaption": "",
            "lowerLimit": "0",
            "upperLimit": "100",
            "lowerLimitDisplay": "Empty",
            "upperLimitDisplay": "Full",
            "numberSuffix": "%",
            "showValue": "1",
            "chartBottomMargin": "55",
            "showValue": "0",
            "refreshInterval": "1",
            "refreshInstantly": "1",
            "cylFillColor": "#b3e6ff",
            "cylradius": "100",
            "cylheight": "300"
        },
```

```
        "value": "70",
        "annotations": {
            "origw": "400",
            "origh": "290",
            "autoscale": "1",
            "groups": [{
                "id": "range",
                "items": [{
                    "id": "rangeBg",
                    "type": "rectangle",
                    "x": "$canvasCenterX-75",
                    "y": "$chartEndY-40",
                    "tox": "$canvasCenterX +50",
                    "toy": "$chartEndY-80",
                    "fillcolor": "#fff25e"
                }, {
                    "id": "rangeText",
                    "type": "Text",
                    "fontSize": "20",
                    "fillcolor": "#333333",
                    "text": "Loading...",
                    "x": "$chartCenterX-40",
                    "y": "$chartEndY-60"
                }]
            }]
        }

    },
```

```
    "events": {
        "rendered": function(evtObj, argObj) {
            evtObj.sender.chartInterval = setInterval(function() {
                getapi();
                if (waterLevel <= 33 && flag)
                {
                    sendEmail();
                    flag=false;
                }
                if(waterLevel > 33){
                    flag=true;
                }
                console.log(waterLevel);
                evtObj.sender.feedData && evtObj.sender.feedData("&value=" + waterLevel);
            }, 1000);
        },
        //Using real time update event to update the annotation
        //showing available volume of Diesel
        "realTimeUpdateComplete": function(evt, arg) {
            var annotations = evt.sender.annotations,
                dataVal = evt.sender.getData(),
                colorVal = (dataVal >= 75) ? "#6caa03" : ((dataVal <= 25) ? "#e44b02" : "#f8bd1b");
            //Updating value
            annotations && annotations.update('rangeText', {
                "text": dataVal + "%"
            });
            //Changing background color as per value
            annotations && annotations.update('rangeBg', {
                "fillcolor": colorVal
```

```
            });

        },
        "disposed": function(evt, arg) {
            clearInterval(evt.sender.chartInterval);
        }
    }
}
);
chartObj.render();
});


const api_url =
    "https://api.thingspeak.com/channels/1358838/feeds.json?api_key=2KJLA50SRMEP7T5I&results=1";
//function to read data from thingspeak cloud
async function getapi() {
    // Storing response
    const response = await fetch(api_url);
    data = await response.json();
    waterLevel = data["feeds"][0]["field1"];

}

//function to send email
function sendEmail() {
    Email.send({
```

```
const api_url =
    "https://api.thingspeak.com/channels/1358838/feeds.json?api_key=2KJLA50SRMEP7T5I&results=1";
//function to read data from thingspeak cloud
async function getapi() {
    // Storing response
    const response = await fetch(api_url);
    data = await response.json();
    waterLevel = data["feeds"][0]["field1"];

}

//function to send email
function sendEmail() {
    Email.send({
    Host: "smtp.gmail.com",
    Username : "gunjan.vinzuda@gmail.com",
    Password : "gunjan123",
    To : "gvinzuda35@gmail.com",
    From : "gunjan.vinzuda@gmail.com",
    Subject : "Water Level is low",
    Body : "The water level in tank is now " + waterLevel + "%",
    }).then(
        message => alert("mail sent successfully")
        // console.log("meail send")
    );
}
```