**SMART DATA LOGGING SYSTEM FOR INDUSTRIES**

**1. ABSTRACT**

The industries monitor the status of the machine continuously and maintain the records of the entire data. This helps the officials to analyze the production factors. This also helps in resolving some of the problems like machine failures, production delay etc. In this project, a system will be created which will capture some of the important parameters of the machines like its temperature, liquid level, etc., All the parameters along with the date and time are stored in Google sheets. Third party services like IFTTT are used for publishing the data to Google sheets. Admin can monitor the entire device parameters and the previous records of data through the Google sheets.

***Keywords:*** ESP8266 Wifi module, LM35 temperature sensor, Ultrasonic sensor, IFTTT service.

**2. INTRODUCTION**

The advancement in the technology in the industries would play a vital role for moving the industry to next level. The projected system does the industrial status monitoring using data logging system. Smart Data logger is an economic and flexible remote data logging system. Functional modules are optional and independent. It ensures high data integrity whenever anyone network fails.

Smart Data logging system helps us in capturing the sensor parameters and publish them to Google spreadsheets with the help of IFTTT service. Here in this system IOT technology is incorporated for publishing the data sensed in the industry.

**3. PROPOSED METHODOLOGY**

To accomplish this, we have to complete all the activities and tasks listed below:

* Configure the IFTTT service
* Create an Applet
* Test your Applet
* Connect the sensors
* Connect temperature sensor
* Connect Ultrasonic sensor
* Develop the code
* Import the libraries
* Define the pin numbers and Configure the network credentials
* Write a function to read sensor data
* Develop the Setup() function
* Develop the Loop() function
* Upload the code and Test the output
* Upload the code
* Test the output

**4. HARDWARE COMPONENTS & SOFTWARE REQUIREMENTS**

The following is the Hardware required to complete this project:

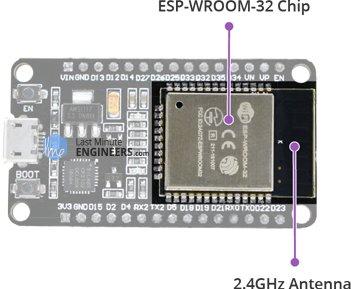
* ESP32 Development Board
* Ultrasonic Sensor
* LM35 Temperature Sensor
* Jumpers
* Breadboard

***a) ESP 32 Module***

The development board equips the ESP-WROOM-32 module containing Tensilica Xtensa Dual-Core 32-bit LX6 microprocessor. This processor is similar to the ESP8266 but has two CPU cores (can be individually controlled), operates at 80 to 240 MHz adjustable clock frequency and performs at up to 600 DMIPS (Dhrystone Million Instructions Per Second).

There’s also 448 KB of ROM, 520 KB of SRAM and 4MB of Flash memory (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays.

The ESP32 Integrates 802.11b/g/n HT40 Wi-Fi transceiver, so it can not only connect to a WiFi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. The ESP32 supports WiFi Direct as well, which is a good option for peer-to-peer connection without the need of an access point. The WiFi Direct is easier to setup and the data transfer speeds are much better than Bluetooth.



***b) Ultrasonic Module***

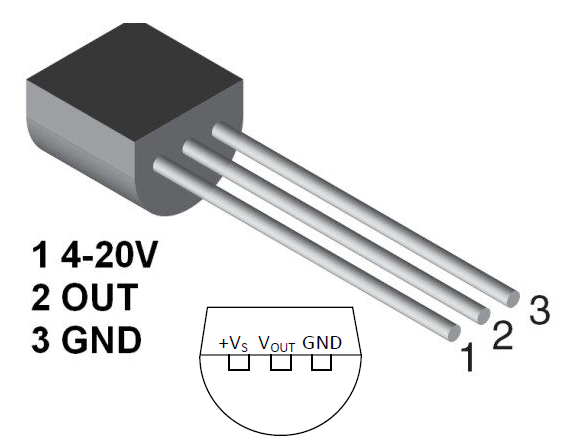
The ultrasonic sensor works on the principle of SONAR and RADAR system which is used to determine the distance to an object. An ultrasonic sensor generates the high-frequency sound (ultrasound) waves. When this ultrasound hits the object, it reflects as echo which is sensed by the receiver.



In ultrasonic module, we have to give trigger pulse, so that it will generate ultrasound of frequency 40 kHz. After generating ultrasound i.e. 8 pulses of 40 kHz, it makes echo pin high. Echo pin remains high until it does not get the echo sound back. So the width of echo pin will be the time for sound to travel to the object and return back. Once we get the time we can calculate distance, as we know the speed of sound.

***c) LM35 Temperature sensor***

LM35 is a temperature sensor that outputs an analog signal which is proportional to the instantaneous temperature. The output voltage can easily be interpreted to obtain a temperature reading in Celsius. The advantage of lm35 over thermistor is it does not require any external calibration. The coating also protects it from self-heating. Low cost (approximately $0.95) and greater accuracy make it popular among hobbyists, DIY circuit makers, and students. Many low-end products take advantage of low cost, greater accuracy and used LM35 in their products. It’s approximately 15+ years to its first release but the sensor is still surviving and is used in any products.

****

***d) Software Requriments***

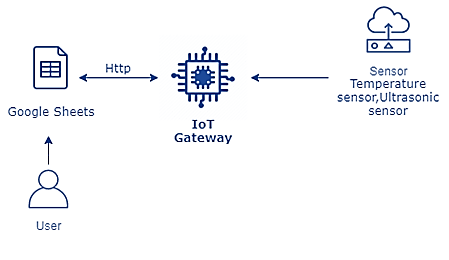
Arduino IDE is to be installed in the PC or laptop to fed the instruction into the controller to function as per the users requirement. Required libraries must be put into the folder for the proper uploading and compiling of the codes.

**5. IFTTT SERVICE**

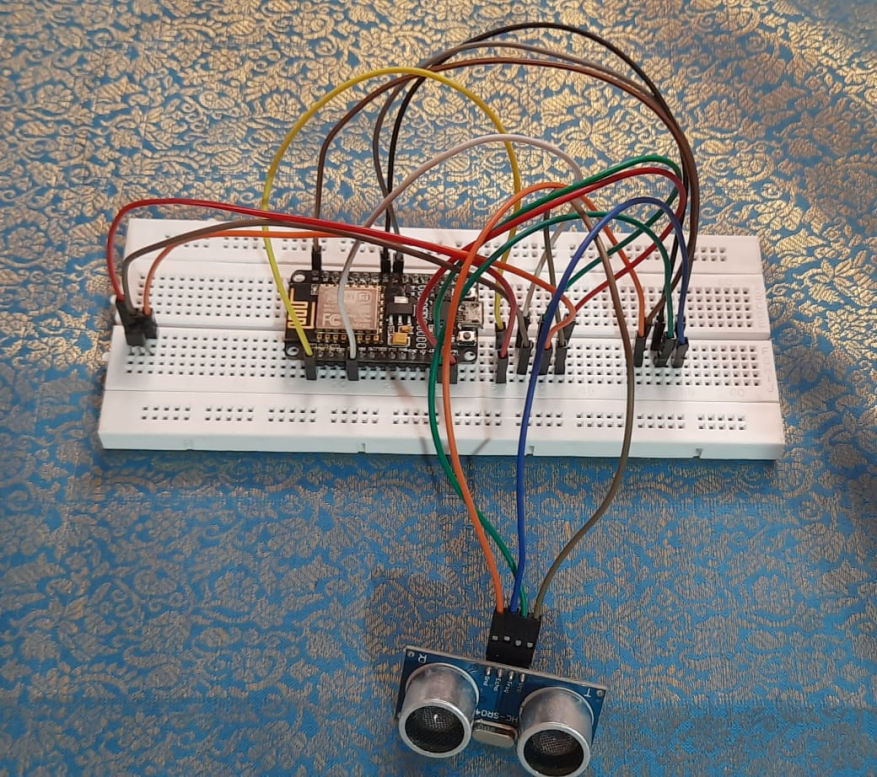
If This Then That (commonly known as IFTTT) is a service that allows a user to program a response to events in the world of various kinds. There is a long list of kinds of events to which IFTTT can respond, all detectable via the Internet. First create the IFTTT service and create an applet to publish data into Google sheets. An Applet connects two or more apps or devices together. It enables you to do something that those apps or devices couldn't do on their own. Applets are composed of triggers and actions. Triggers tell an Applet to start, and actions are the end result of an Applet run. So here the trigger is receiving the web request and the action is publishing the sensor data to Google spreadsheets.

**6. BLOCK DIAGRAM OF THE SYSTEM**

In this project, a system will be created which will capture some of the important parameters of the machines like its temperature, liquid level, etc., All the parameters along with the date and time are stored in Google sheets. Third party services like IFTTT are used for publishing the data to Google sheets. Admin can monitor the entire device parameters and the previous records of data through the Google sheets.



**7. HARWARE SETUP**



**8. CODE TO FUNCTION THE SYSTEM**

#include <ESP8266WiFi.h>

const char\* ssid = "Galaxy M30sFFEF";

const char\* password = "balajiwifi";

const int analogIn = A0;

int trigPin=D5;

int echoPin=D6;

int RawValue= 0;

double Voltage = 0;

double tempC = 0;

double tempF = 0;

int distance,duration;

// Replace with your unique IFTTT URL resource

const char\* resource = "/trigger/datalogger/with/key/bvS4JsuzMDNbQGrrtAk\_nb";

//https://maker.ifttt.com/trigger/datalogger/with/key/lwc9YxLFtBzVZVhPV3dFrKiG0DmwVOytHudBbzNO2lb

// Maker Webhooks IFTTT

const char\* server = "maker.ifttt.com";

// Time to sleep

uint64\_t uS\_TO\_S\_FACTOR = 1000000; // Conversion factor for micro seconds to seconds

// sleep for 30 minutes = 1800 seconds

uint64\_t TIME\_TO\_SLEEP = 1800;

void sensorReadings(){

RawValue = analogRead(analogIn);

Voltage = (RawValue / 2048.0) \* 3300; // 5000 to get millivots.

tempC = Voltage \* 0.1;

tempF = (tempC \* 1.8) + 32; // convert to F

Serial.print("Raw Value = " ); // shows pre-scaled value

Serial.print(RawValue);

Serial.print("\t Temperature in C = ");

Serial.print(tempC,1);

Serial.print("\t Temperature in F = ");

Serial.println(tempF,1);

delay(500);

digitalWrite(trigPin,LOW);

delay(1000);

digitalWrite(trigPin,HIGH);//generate one pulse by making trig pin high and low

delay(1000);

digitalWrite(trigPin,LOW);

duration=pulseIn(echoPin,HIGH);

Serial.println("duration value");

Serial.println(duration);

distance=(duration\*0.343)/2;

Serial.println("distance value");

Serial.println(distance);

}

void setup() {

Serial.begin(115200);

delay(2000);

pinMode(trigPin,OUTPUT);

pinMode(echoPin,INPUT);

initWifi();

// esp\_sleep\_enable\_timer\_wakeup(TIME\_TO\_SLEEP \* uS\_TO\_S\_FACTOR);

Serial.println("Going to sleep now");

// esp\_deep\_sleep\_start();

}

void loop() {

makeIFTTTRequest();

delay(1000);

delay(1000);

}

// Establish a Wi-Fi connection with your router

void initWifi() {

Serial.print("Connecting to: ");

Serial.print(ssid);

WiFi.begin(ssid, password);

int timeout = 10 \* 4; // 10 seconds

while (WiFi.status() != WL\_CONNECTED && (timeout-- > 0)) {

delay(250);

Serial.print(".");

}

Serial.println("");

if (WiFi.status() != WL\_CONNECTED) {

Serial.println("Failed to connect, going back to sleep");

}

Serial.print("WiFi connected in: ");

Serial.print(millis());

Serial.print(", IP address: ");

Serial.println(WiFi.localIP());

}

// Make an HTTP request to the IFTTT web service

void makeIFTTTRequest() {

sensorReadings();

Serial.print("Connecting to ");

Serial.print(server);

WiFiClient client;

int retries = 5;

while (!!!client.connect(server, 80) && (retries-- > 0)) {

Serial.print(".");

}

Serial.println();

if (!!!client.connected()) {

Serial.println("Failed to connect...");

}

Serial.print("Request resource: ");

Serial.println(resource);

// Temperature in Celsius

String jsonObject = String("{\"value1\":\"") + tempC + "\",\"value2\":\"" + distance

+"\"}";

client.println(String("POST ") + resource + " HTTP/1.1");

client.println(String("Host: ") + server);

client.println("Connection: close\r\nContent-Type: application/json");

client.print("Content-Length: ");

client.println(jsonObject.length());

client.println();

client.println(jsonObject);

int timeout = 5 \* 10; // 5 seconds

while (!!!client.available() && (timeout-- > 0)) {

delay(100);

}

if (!!!client.available()) {

Serial.println("No response...");

}

while (client.available()) {

Serial.write(client.read());

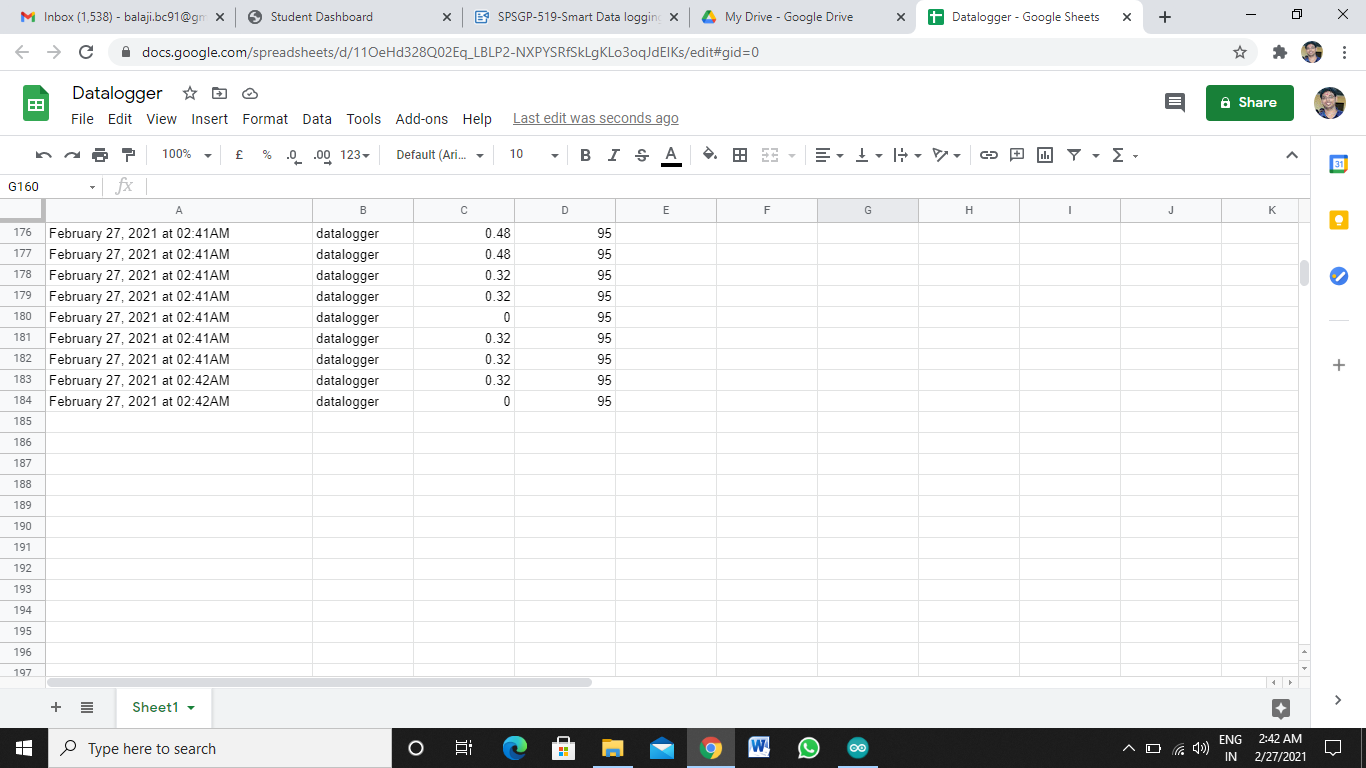
}

Serial.println("\nclosing connection");

client.stop();

}

**9. OUTPUT OBTAINED IN GOOGLE SHEET**

****

**10. CONCLUSION**

Thus the system created capture some of the important parameters of the machines like its temperature, liquid level, etc., All the parameters along with the date and time are published in the Google sheets. Third party services like IFTTT are used for publishing the data to Google sheets. Admin can monitor the entire device parameters and the previous records of data through the Google sheets. Thus the system works effectively which helps to maintain the database and helps to find out the issue if occurred in the industry during the process.

**11. REFERENCES**

[1] K. C. Lee and H.-H. Lee, “Network-based Fire-Detection System via Controller Area Networkfor Smart Home Automation”, 0098 3063/04/$20.00 © 2004 IEEE

[2] Jadsonlee Silva Sá and JaidilsonJó da Silva, “Monitoring of Temperature Using Smart Sensors Based on CAN Architecture”, 0-7695-2283-1/05 $20.00 © 2005

[3] Zhu Qishen and Zhu Dongmei, “Distributed Remote Temperature Monitoring and AcquisitionSystem Based on CAN Bus”, 978-1-4244-4758-9/10/$26.00 © 2010

[4] Minu A Pillai and SrideviVeerasingam, “CAN Based Smart Sensor Network for Indoor Air Quality Monitoring,” 978-1- 4244-5540-9/10/$26.00 ©2010

[5] Nhivekar, G. S. &Mudholker, R. R. 2011. Data Logger and Remote Monitoring System for Multiple Parameter Measurement Applications. e-Journal of Science & Technology, vol.3, no.6,pp. 55-62

[6] A.Manikandan, J.Sakthivel, “Solid Waste Bin Management System”, Journal of Advanced Research in Dynamical and Control Systems, ISSN:1943-023X , pp 29-33, 2017.

[7] K.Bommaraju, A. Manikandan, S.Ramalingam, “Aided System for Visually Impaired People in Bus Transport using Intel Galileo Gen-2” International Journal of Vehicle Structures & Systems, Online ISSN:0975-3540, Volume 9, Issue 2, pp 110-112,2017.

[8] NirmalKumar p,A.Manikandan, “Zigbee Based Online Air Pollution Monitor” Journal of Chemical and Pharmaceutical Sciences, ISSN:0974-2115, pp 230-232, Feb 2017.

[9] A.Manikandan, S.Tamilselvan, “RFID Based Voice Bank Alert System for Blind People” Journal of Chemical and Pharmaceutical Sciences, ISSN:0974-2115, pp 308-310, Feb 2017.

[10] A.Manikandan, J.Sakthivel, “IoT Based Real Time Traffic Analyzer” Journal of Chemical and Pharmaceutical Sciences, ISSN:0974-2115, pp 139-142, Dec 2016.

[11] Ling, N. & Mei-Xia, D. 2015. Design of a Remote Data Monitoring System based on Sensor Network. International Journal of Smart Home, vol. 9, no. 5, pp. 23-30.