HAZARDOUS AREA MONITORING FOR INDUSTRIAL PLANT POWERED BY IOT

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A PROJECT REPORT

SYED AKRAM (718019L143)
ABDUL ZUBAIR(718020L401)
CHANDEESH KS(718020L40)
NISHANTH RAJ T(718020L409)

BACHELOR OF ENGINEERING IN ELECTRONICS AND COMMUNICATION ENGINEERING

NOVEMBER 2022

ABSTRACT

The objective of the Internet of Things (IoT), a model industry, is to link "things," "people," and "machines" to the internet. Global modernization and automation are on the rise, and IoT-based industrial monitoring systems are leading the way. For the safety and effectiveness of the products, it is critical to evaluate the state of the industry. This work aims to build an IoT-based industrial monitoring system with smart sensors. The Mobile application allows users to check on status from anywhere on the earth thanks to the integration of big data. Data analysis has been optimised to make IoT monitoring simpler. Manufacturing industries or hazardous power plants might profit from the proposed technology.

TABLE OF CONTENTS

1.INTRODUCTION:

Many industries are influenced by the Internet of Things (IoT), which has an effect on daily life as well as business operations. The Internet of Things (IoT) refers to the transition from computer networks to an object network where each element of daily social and professional life is connected. Numerous fatalities, serious injuries, and catastrophic damages that have disrupted people's lives and caused suffering for present and future generations have occurred as a result of accidents, shortcomings, or plain negligence on the part of industry authorities. This proposal proposes a cutting-edge checking approach based on the Internet of Things to prevent any similar calamity in the future (IoT). This building project establishes a mechanical observation framework that can detect abnormal quantities of gases, such as hydrogen, butane, LPG, and carbon monoxide, which could cause an explosion.It additionally shows air volume. The business may also check the humidity levels in addition to keeping an eye on the temperature and clearing up any pollutants.The safety of the industry is guaranteed by the integration of data from several sensors. The system runs consistently and dependably. It is the most effective and accountable way to keep an eye on hardware security.

2.OBJECTIVE:

Since temperature variations, which are a physical aspect of the environment, have an effect on the operations of various industrial equipment, monitoring temperature variations is particularly crucial.

· A microcontroller chip is built inside the computer to regulate various settings, and a system monitors the collection of real-time data. Values from different parameters are combined and displayed on LCD.

· The Arduino has a collection of all the code burned into it. A specific parameter, such as air, temperature, pressure, or humidity, is represented by each code. Intelligent industrial remote monitoring of the power system, intelligent furniture monitoring, intelligent warehouse monitoring, etc. may all be implemented using the systems platform. The user is reassured by this of the system's dependability and stability. It has positive social features and is the most efficient and cost-effective way to monitor equipment safety.

· It detects temperature changes, detects smoke, detects flames, etc., and communicates that information to the control station via an Android app.

· To pinpoint the precise location of fire threats that have occurred, sensors were installed in the prototype at three different locations.

3. IDEATION PHASE

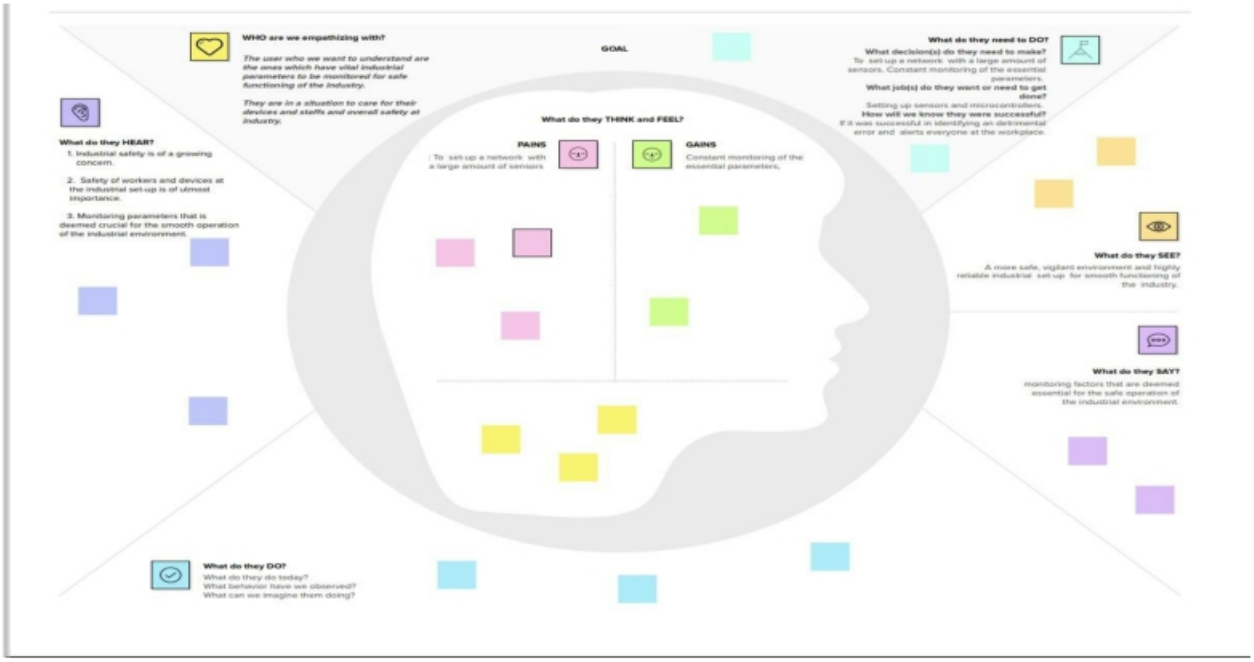
3.1 Literature Survey:

The first step toward safety is monitoring. There are numerous industries that use hazardous chemical gases in daily life, and their employees are frequently exposed to these gases.Such gases have an unanticipated and a significant influence on properties and human life. To keep

these things from happening, an automated system for detecting and alerting toxic gasses is built.The suggested method consists of a monitoring and notification system powered by the Internet of Things (IoTs). Gasses like hydrogen sulphide, which is poisonous and combustible, are present in this.

A smart monitoring system for the detection of poisonous gas residues, reduced oxygen concentrations, and flammable gas residues is built. The suggested method intends to increase worker safety by decreasing the risk of fires and explosions when they are doing maintenance on or inspecting gas storage facilities. The monitoring system is built on small battery-operated wearable sensor nodes that have sensors for oxygen, hazardous gasses, and flammable LPG compounds. By including an intrusion detection system, which forbids unauthorized entry to safety-critical locations to prevent mishaps, the proposed system can help boost plant safety. The sensor nodes use a BLE to identify users and grant access to restricted areas while transmitting data to a remote server over a LoRa low power radio channel.

3.2Empathy Map:



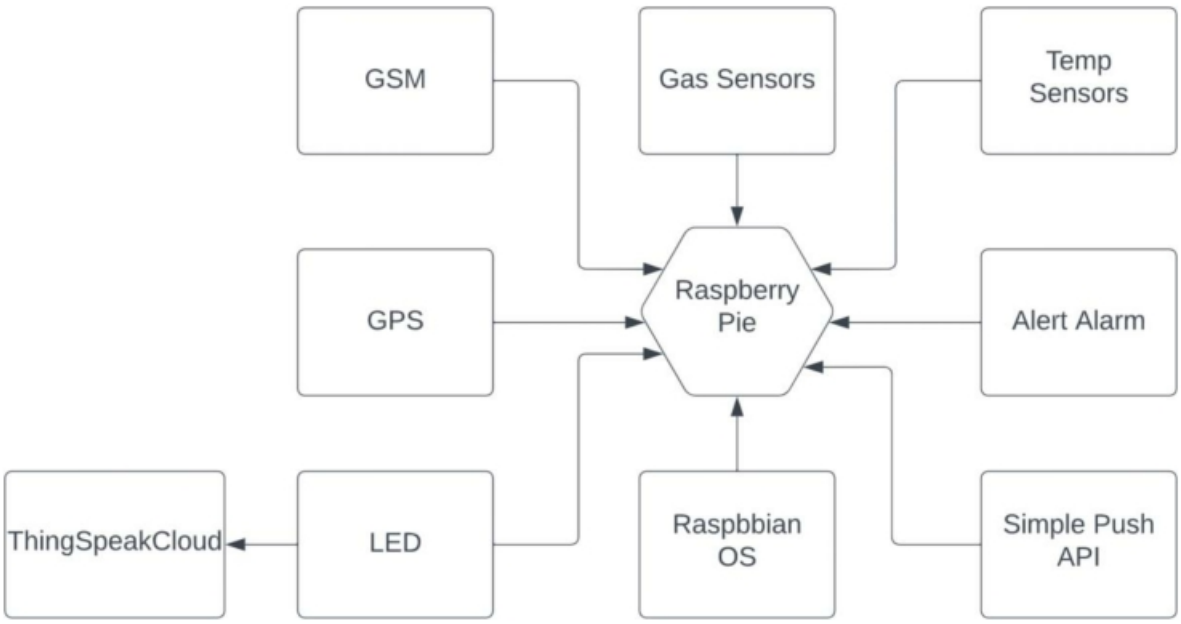
4.PROJECT PHASE 1

4.1 Proposed Solution:

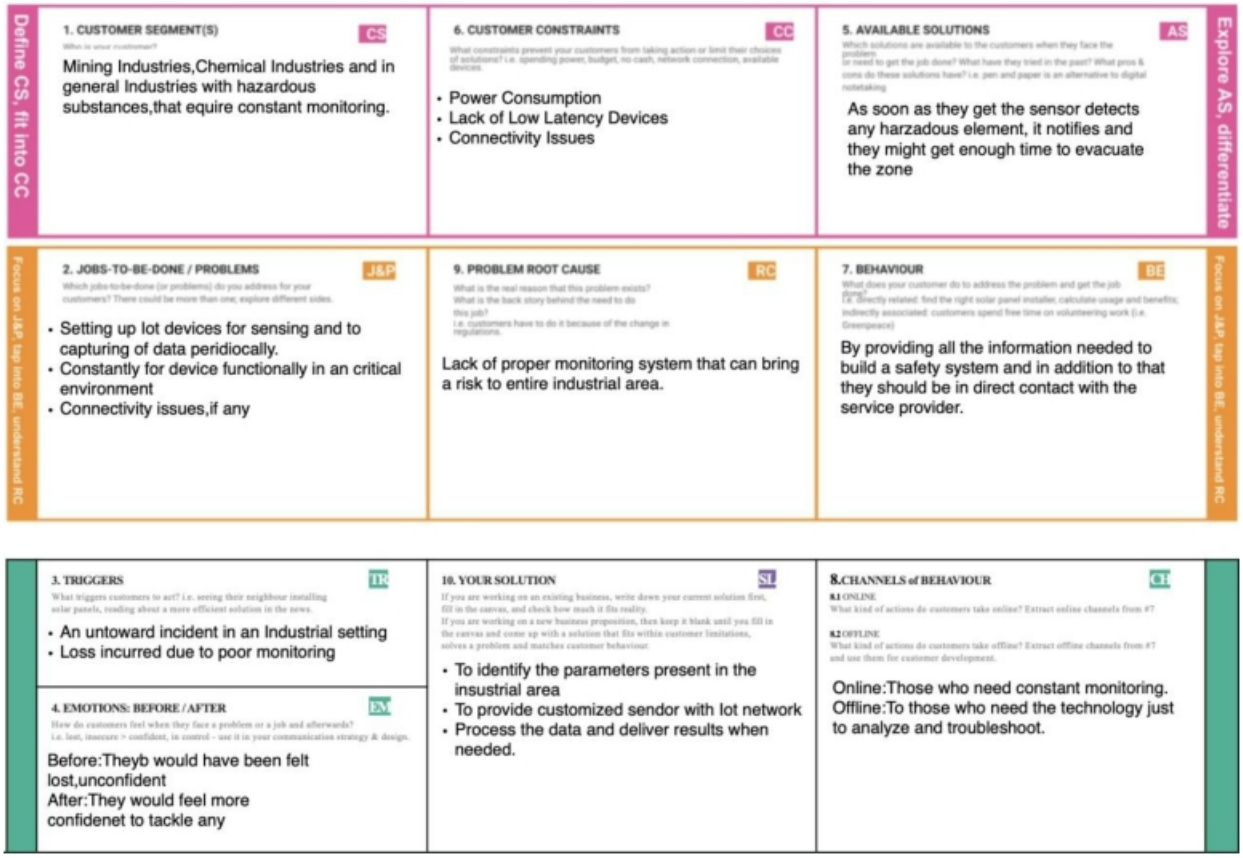
S.No	Parameter	Description
1	Problem Statement (Problem to be solved)	Hazardous Area Monitoring for Industrial Plant powered by IoT
2	Idea / Solution description	The project focuses on the necessity of monitoring hazardous regions in industrial plants. The only places that include both hazardous and nonhazardous sections are industrial plants. Occasionally, it's crucial to monitor the dangerous locations in industrial buildings. If the destruction in these regions poses a risk to persons or property, it should be avoided. Monitoring these locations can therefore make it simpler to monitor hazardous areas. Hazardous regions may have smart gadgets included that can assist in spotting any suspicious activity that might take place there.
3	Novelty / Uniqueness	* A hazardous zone is any location with an environment that contains or may contain explosive or flammable gasses, vapor, or dust. When installing equipment, these regions are meticulously examined with condition monitoring to reduce the risk to people and property. Effective monitoring of equipment functioning in these circumstances is essential to preventing problems before they arise. Unlike most sectors, these problems don't just cause downtime, but present a significant safety risk.* To prevent downtime, perform regular maintenance, and lower the risk of failure, condition monitoring is crucial in industrial operations. Due to a lack of affordable and simple installation options, as well as the frequently difficult situations in which this equipment exists, remote condition monitoring has thus been restricted in hazardous locations. Equipment utilized in underwater applications or on offshore

		operations, for instance, cannot be easily or routinely inspected.
4	Social Impact / Customer Satisfaction	1) To stop pollution 2) Continuous plant monitoring; 4) Computerized detection 5) Fantastic client service
5	Business Model (Revenue Model	Raspberry -Pi 3 Temperature Sensor - DS18B20 Gas Sensor - MQ 5/9 Breadboard Raspbian OS (Running on Rpi-3) Simple push API Thing speak Cloud Platform
6	Scalability of the Solution	This system is deployable in numerous industrial mines, underground factories, and metal factories for automated welding, refineries, and even production lines for heavy parts. It's beneficial to create a productive and safe work environment while also opening new strategies to raise the safety standards of the location.

4.2 Solution Architecture:

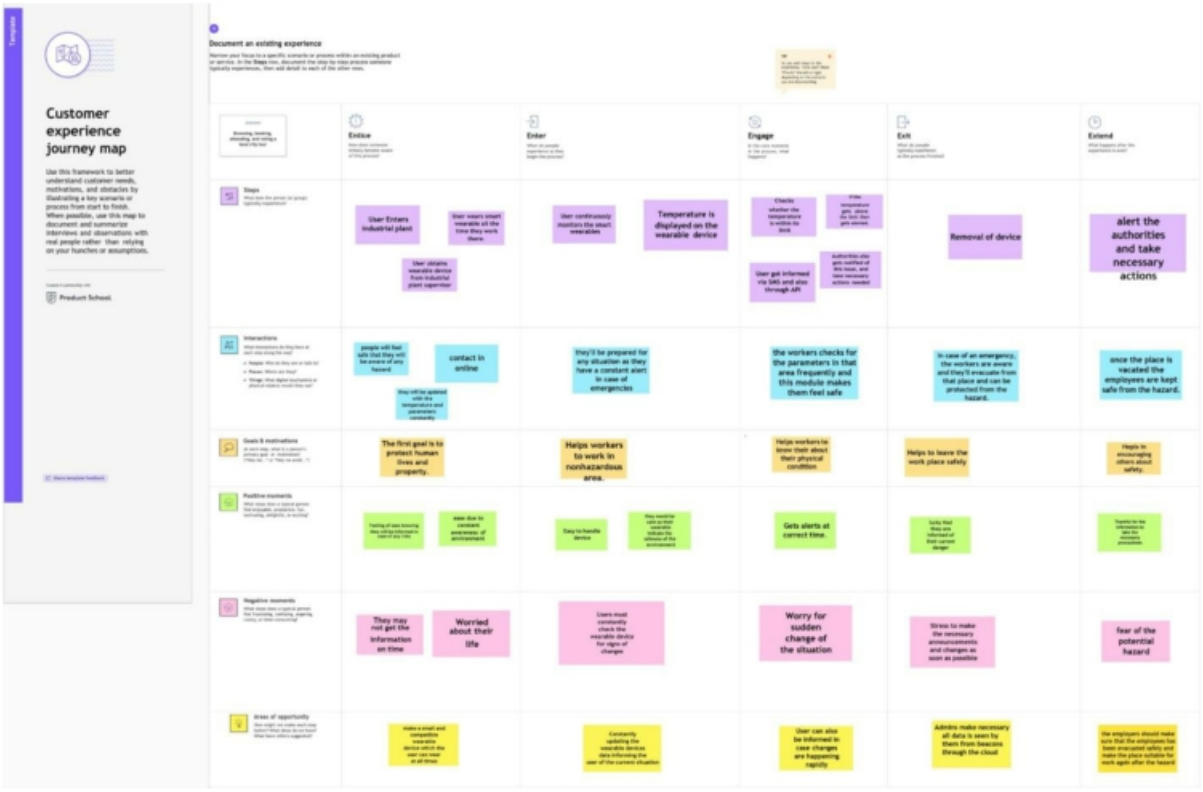


4.3 Solution Fit:



5. Project Design Phase 2

5.1 Customer Journey map:



5.2 Solution Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Temperature sensors	To detect the temperature of a particular zone.
FR-2	Beacons	To broadcast the information.
FR-3	Smart wearables	To notify the workers about the parameters.

FR-4	Alarm	To alert the workers nearby.
FR-4	Mobile App	To alert the workers if the temperature goes beyond the limit.
FR-5	Cloud Storage	To store and access the data.

Non-functional Requirements:

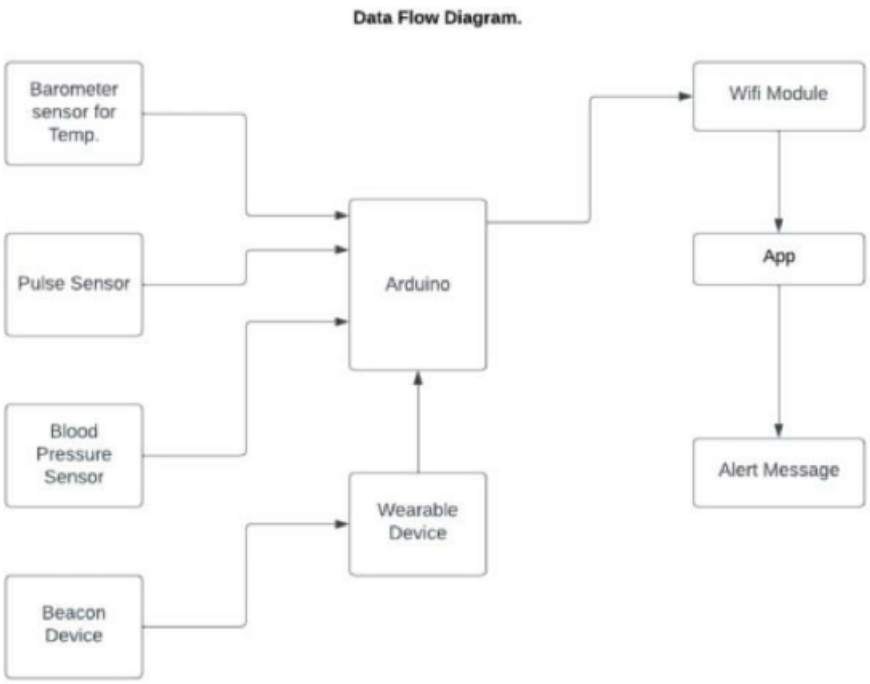
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	User-friendly wearable devices.
NFR-2	Security	The primary thought of installing the devices in the industry is for security for the workers.
NFR-3	Reliability	Data are saved in the secured server so they don’t provide any loopholes for the hackers
NFR-4	Performance	There wont be any sort of server crash.

NFR-5	Availability	Information is available all through out wearable devices and mobile application
NFR-6	Scalability	Easily accessible for a large group of peoples.

5.3 Data Flow Diagrams:

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



User Experience:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Worker in an industrial plant	Registration	USN-1	As a user, I can wear the device to see the temperature and will receive the alerts to the device through app once I have registered in the application.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email As soon as I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Whatsapp/ Telegram/Gmail.	I can register & access the dashboard.	Low	Sprint-2
		USN-4	As a user, I can log into the application by entering email & password	I can receive confirmation email & click confirm	Medium	Sprint-1
Worker in industry	Login	USN 1	As a user, I can wear the device comfortably and I can view the changes in temperature.			
		USN 2	As a user, I can receive the alerts correctly to the mobile through app.	I can receive confirmation email		
Worker in mining industry	Login	USN 1	As a user, I can view the temperature through wearable device even though there is no proper signal.			
Administrator	Login	USN 1	As a user, I can take necessary actions once the alerts are received according to the problem	I can access my account through application	High	Sprint-1
		USN 2	I can check the temperature conditions through app and wearable device.		Medium	Sprint-1

5.4Technology Stack

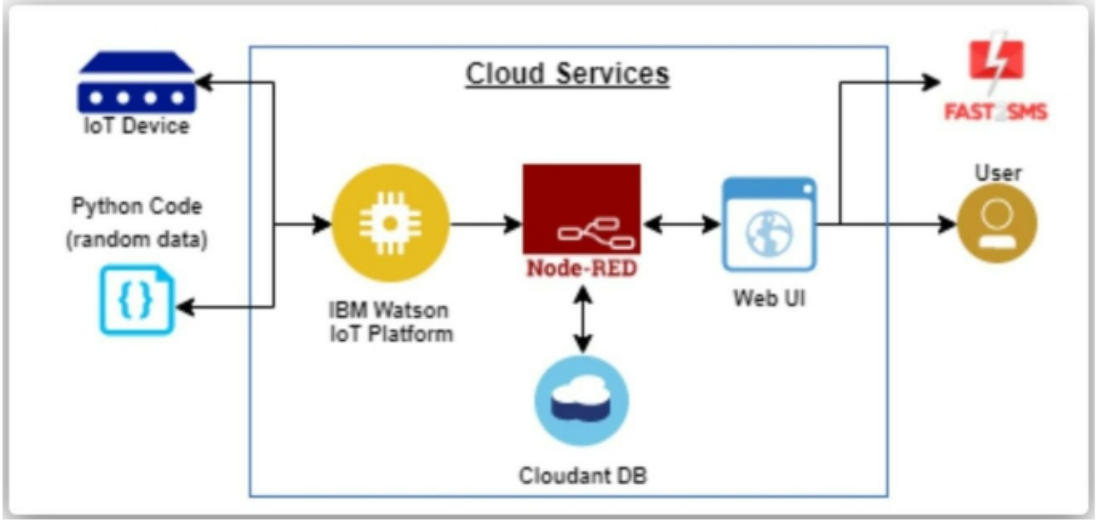


Table for Technology Stack:

S.No	Component	Description	Technology
1.	User Interface	Web UI, Mobile App, SMS service and Wearable devices	Node-RED, Fast SMS and MIT App inventor
2.	Application Logic-1	Getting input from smart beacons	Embedded C and Python
3.	Application Logic-2	Process data in cloud	IBM Watson IOT platform, Cloud-ant DB and NodeRED
4.	Application Logic-3	Display data to the user	Web UI, Fast SMS and Mobile

			application
5.	Database	Real time database	Cloud-ant DB
6.	Cloud Database	Database Service on Cloud	IBM Cloud-ant
7.	Smart Beacon	To monitor the area and update the stats in the cloud	Node MCU and Sensors
8.	External API-1	To send SMS to user	Fast SMS API
9.	External API-2	Language for the website is written to be dynamic	Google translate API
10.	External API-3	To access time	World time API
11.	Infrastructure (Server / Cloud)	Application Deployment on Cloud	IBM Cloud

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The Node-RED open-source frameworks are used to build the web application as well as to communicate with the mobile application and to handle alert SMS	Node-RED framework
2.	Security Implementations	Need to secure the datas so that the hackers dont get access to that	Encryptions
3.	Scalable Architecture	The 3 – tier architecture used with a separate user interface, application tier and data tier makes it easily scalable	IBM wat-son studio
4.	Availability	The web application is highly available as it is deployed in cloud	IBM cloud
5.	Performance	The performance of the website is improved with caching and security	IBM cloud internet services

6.PROJECT PLANNING PHASE

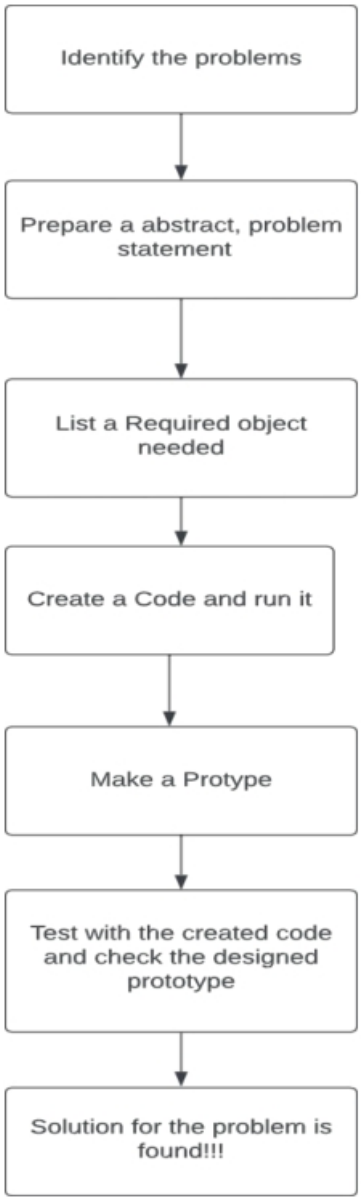
6.1 prepare Milestone and Activity List

TITLE	DESCRIPTION	DATE
Literature survey and Information gathering	Literature survey in the selected topic &gathering information by referring the technical papers, research publications etc…	05 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvasto capture the user Pains & Gains, Prepare list of problem statements.	11 SEPTEMBER 2022
Ideation	List by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	16 SEPTEMBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact,scalability of solution, etc.	21 SEPTEMBER 2022

Problem Solution Fit	Prepare problem - solution fit document.	24 SEPTEMBER 2022
Solution Architecture	Prepare solution architecture document.	30 SEPTEMBER 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application.	05 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	8 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit forreview.	9 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	10 OCTOBER 2022
Prepare Milestone & ActivityList	Prepare the milestones &activity list of the project.	22 OCTOBER 2022

Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS...

6.2 Sprint Delivery Plan



7. Project Development Phase

7.1 Delivery of Sprint –1:

```
#include <dht.h> #define
dht_apin A0 #define mqt_apin
A1 dht DHT;
int senvalue; void
setup()

{
 Serial.begin(9600); Serial1.begin(9600);
   delay(500);

}
void loop()
{
DHT.read11(dht_apin);
   senvalue = analogRead(mqt_apin); pin
     1(MQ135)
CodeSerial.print("Current humidity = ");
Serial.print(DHT.humidity); Serial.print("% ");
Serial.print("temperature = ");
Serial.print(DHT.temperature); Serial.println("C ");
Serial.print("AirQua="); Serial.print(senvalue, DEC);
Serial.println(" PPM"); Serial1.println("H T A");
Serial1.println(DHT.humidity);
```

```
Serial1.println(DHT.temperature); Serial1.println(senvalue,
DEC); delay(100); }
```

7.2 Project Development- Delivery of Sprint –2:

```
 #include <WiFi.h>
 #include
 <PubSubClient.h>
 #include <DHT.h>
 WiFiClient wifiClient;
 String value;

#define DHTTYPE
DHT11 #define
DHTPIN 9

DHT dht(DHTPIN, DHTTYPE);
#define ORG "v6wg8x"
 #define DEVICE_TYPE
 "nodeMcu" #define
 DEVICE_ID "NodeMCU"
 #define TOKEN "123456789"

#define speed 0.034
 void callback(char* topic, byte* playload, unsigned int payloadLength); char server[] =
 ORG ".messaging.internetofthings.ibmcloud.com";
```

```
char publishTopic[] = "iot-2/evt/Data/fmt/json"; char
topic[] = "iot-2/cmd/test/fmt/String"; char authMethod[]
= "use-token-auth";

char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
  PubSubClient client(server, 1883, callback , wifiClient); void
  publishData();

String
command;
String data =
""; long
duration; float
distance; void
setup()

{

      Serial.begin(115200);

      dht.begin
      ();
      wifiConn
      ect();
      mqttCon
      nect();
```

```
}

 void
 loop()
 { publish
 D ata();
 delay(700
 );
   if (!client.loop())
   { mqttConnect();

     }

}
void wifiConnect() {
 Serial.print(“Connecting: "); Serial.print("Wifi");
 WiFi.begin("SSID","Passord");

while (WiFi.status() != WL_CONNECTED)
{ delay(700);

Serial.print(".");

     }
 Serial.print("WiFi connected, IP address: ");
 Serial.println(WiFi.localIP());
}
void mqttConnect() {
```

```
if (!client.connected()) {
  Serial.print("Reconnecting MQTT client to "); Serial.println(server); while
  (!client.connect(clientId, authMethod, token)) {
                                    Serial.print(".");

    delay(700);
          }
      initManagedDevi
      ce();
      Serial.println();

      }
}

void initManagedDevice() {
    if (client.subscribe(topic)) {
    Serial.println("IBM subscribe to cmd OK");
      } else {
    Serial.println("subscribe to cmd FAILED");
      }
}

void publishData()
{
```

```
    int sensorValue = analogRead(34); //MQT 135 connected to GPIO 34 (Analog
  ADC1_CH6)
    Serial.print("AirQua=");
    Serial.print(sensorValue, DEC);
    Serial.println(" PPM");

    float humid = dht.readHumidity();
    float temp = dht.readTemperature(true);
    float airQty = sensorValue/4095;
    String payload = "{\"Temperature\":";
    payload += temp;

    payload += "}";
        if (client.publish(publishTopic, (char*) payload.c_str())) {
            Serial.println("Publish OK");

    }
    payload = "{\"Air Quality\":";
    payload += airQty;

    payload += "%}";
        if (client.publish(publishTopic, (char*) payload.c_str())) {
            Serial.println("Publish OK");

    }
}
```

```
void callback(char* subscribeTopic, byte* payload, unsigned int
payloadLength) {
    Serial.print("callback invoked for topic:");
    Serial.println(subscribeTopic);
        for (int i = 0; i < payloadLength; i++) {
            distance += (char)payload[i];


    }
    Serial.println("data:" + value); if
    (value == "lighton") {

        Serial.println(value);


    }
    value = "";
}
```

7.3 Project Development- Delivery of Sprint –3:

```
#include <WiFi.h> #include
<PubSubClient.h>

#include <DHT.h> WiFiClient wifiClient; String data3;

#define DHTTYPE DHT11
#define DHTPIN 4

#define MQTPIN 34

DHT dht(DHTPIN, DHTTYPE);

#define ORG "v6wg8x"
```

```
  #define DEVICE_TYPE "projectFinal" #define DEVICE_ID
  "FinalDeliverable" #define TOKEN "A1ymH))p*JB&iMWNpY" #define
  speed 0.034
  void callback(char* topic, byte* playload, unsigned int payloadLength); char
  server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/Data/fmt/json";

  char topic[] = "iot-2/cmd/test/fmt/String"; char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

  PubSubClient client(server, 1883, callback , wifiClient); void publishData();

String command; String data = ""; long duration;
float dist;

void setup()

{

    Serial.begin(115200); dht.begin(); wifiConnect(); mqttConnect();

}


    void loop() { publishData(); delay(500);


      if (!client.loop()) { mqttConnect();}


void wifiConnect() {

    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("JerroldWi-Fi","75779901");
      while (WiFi.status() != WL_CONNECTED) { delay(500);
      Serial.print(".");

    }
```

```
    Serial.print("WiFi connected, IP address: ");
  Serial.println(WiFi.localIP());

}


void mqttConnect() {

      if (!client.connected()) { Serial.print("Reconnecting MQTT
        client to ");
      Serial.println(server); while (!client.connect(clientId, authMethod, token)) {

        Serial.print("."); delay(500);

      }
      initManagedDevice(); Serial.println();
    }
}


void initManagedDevice() {

        if (client.subscribe(topic)) { Serial.println("IBM subscribe to cmd OK");

    } else {

      Serial.println("subscribe to cmd FAILED");
    }
}


void publishData()

{

    int sensorValue = analogRead(MQTPIN); //MQT 135 connected to GPIO 34 (Analog
  ADC1_CH6)

    Serial.print("AirQua="); Serial.print(sensorValue, DEC); Serial.println(" PPM");

    float humid = dht.readHumidity();
```

```
    float temp = dht.readTemperature(true); String payload = "{\"Humidity\":";
    payload += humid;

    payload += "}";

        if (client.publish(publishTopic, (char*) payload.c_str())) { Serial.println("Publish
          OK");

    }

    payload = "{\"Temperature\":"; payload += temp; payload += "}";
        if (client.publish(publishTopic, (char*) payload.c_str())) { Serial.println("Publish
          OK");

    }

    payload = "{\"Air Quality\":"; payload += String(sensorValue); payload += "}";
        if (client.publish(publishTopic, (char*) payload.c_str())) { Serial.println("Publish
          OK");

    }

 void callback(char* subscribeTopic, byte* payload, unsigned int payloadLength) {
    Serial.print("callback invoked for topic:");
    Serial.println(subscribeTopic);
        for (int i = 0; i < payloadLength; i++) { dist += (char)payload[i];

    }

      Serial.println("data:" + data3); if (data3 == "lighton") { Serial.println(data3);

    }

    data3 = "";
}
```

Output:

```
Published Temperature=98 C Noise:61 db Gas_leakage:63 J/Kg Radiation:45 rad  to IBM Watson
Published Temperature=19 C Noise:4 db Gas_leakage:97 J/Kg Radiation:73 rad  to IBM Watson
Published Temperature=70 C Noise:0 db Gas_leakage:85 J/Kg Radiation:64 rad  to IBM Watson
Published Temperature=74 C Noise:61 db Gas_leakage:54 J/Kg Radiation:97 rad  to IBM Watson
Published Temperature=47 C Noise:77 db Gas_leakage:50 J/Kg Radiation:91 rad  to IBM Watson
Published Temperature=78 C Noise:0 db Gas_leakage:33 J/Kg Radiation:27 rad  to IBM Watson
Published Temperature=17 C Noise:6 db Gas_leakage:99 J/Kg Radiation:78 rad  to IBM Watson
Published Temperature=7 C Noise:38 db Gas_leakage:98 J/Kg Radiation:69 rad  to IBM Watson
Published Temperature=5 C Noise:79 db Gas_leakage:91 J/Kg Radiation:50 rad  to IBM Watson
Published Temperature=20 C Noise:35 db Gas_leakage:21 J/Kg Radiation:4 rad  to IBM Watson
Published Temperature=35 C Noise:73 db Gas_leakage:11 J/Kg Radiation:27 rad  to IBM Watson
Published Temperature=61 C Noise:73 db Gas_leakage:55 J/Kg Radiation:68 rad  to IBM Watson
Published Temperature=99 C Noise:76 db Gas_leakage:62 J/Kg Radiation:32 rad  to IBM Watson
Published Temperature=40 C Noise:28 db Gas_leakage:1 J/Kg Radiation:97 rad  to IBM Watson
Published Temperature=10 C Noise:24 db Gas_leakage:83 J/Kg Radiation:76 rad  to IBM Watson
Published Temperature=50 C Noise:18 db Gas_leakage:95 J/Kg Radiation:95 rad  to IBM Watson
Published Temperature=60 C Noise:21 db Gas_leakage:43 J/Kg Radiation:0 rad  to IBM Watson
Published Temperature=60 C Noise:25 db Gas_leakage:5 J/Kg Radiation:3 rad  to IBM Watson
Published Temperature=51 C Noise:40 db Gas_leakage:18 J/Kg Radiation:19 rad  to IBM Watson
Published Temperature=0 C Noise:8 db Gas_leakage:91 J/Kg Radiation:58 rad  to IBM Watson
Published Temperature=41 C Noise:17 db Gas_leakage:90 J/Kg Radiation:95 rad  to IBM Watson
Published Temperature=5 C Noise:30 db Gas_leakage:40 J/Kg Radiation:13 rad  to IBM Watson
Published Temperature=29 C Noise:97 db Gas_leakage:9 J/Kg Radiation:46 rad  to IBM Watson
Published Temperature=6 C Noise:84 db Gas_leakage:64 J/Kg Radiation:80 rad  to IBM Watson
Published Temperature=54 C Noise:73 db Gas_leakage:73 J/Kg Radiation:46 rad
```

7.4 Project Development- Delivery of Sprint –4:

```
#include<WiFi.h>
#<PubSubClient.h>
#define DHTPIN 15
DHTTYPE DHT22

#define LED 2


DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht
connected void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-------credentials of IBM Accounts------



#define ORG "$$$$$$"//IBM ORGANITION ID

#define DEVICE_TYPE "nodeMCU"//Device type mentioned in ibm watson IOT
Platform #define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT
Platform #define TOKEN "12345678"

//Token

String data3; float h, t;
```

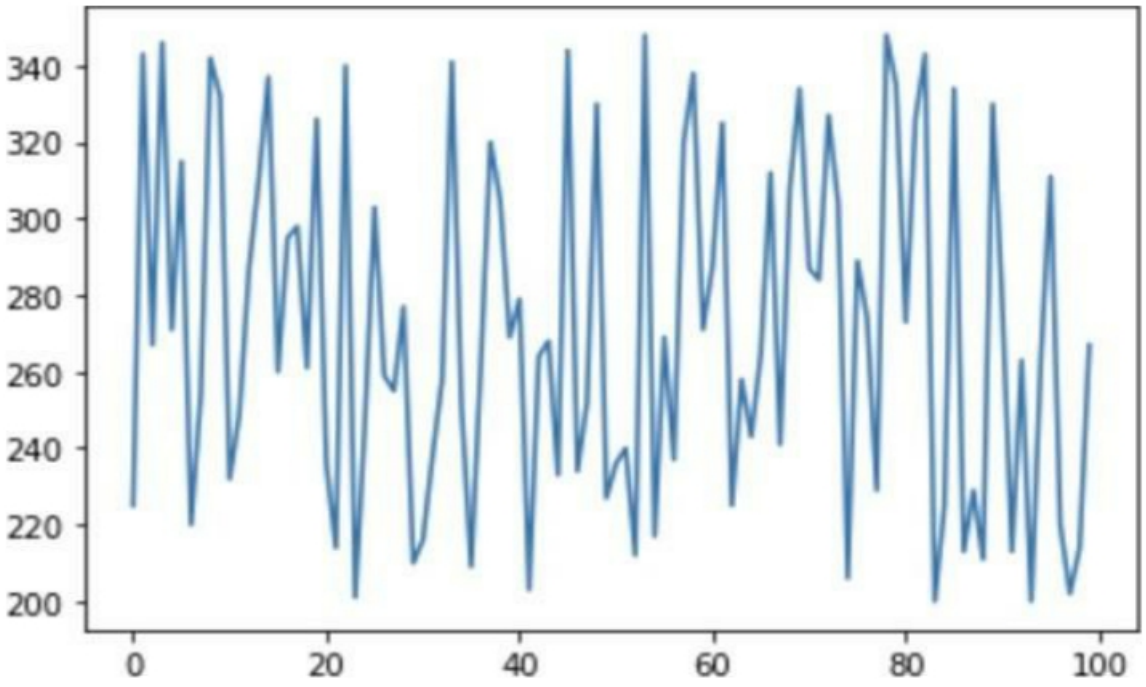
```
//-------- Customise the above values -------- char server[] = ORG
".messaging.internetofthings.ibmcloud.com";// Server Name char publishTopic[]

= "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which
data to be send

char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING

char authMethod[] = "use-token-auth";// authentication method char token[] = TOKEN; char
clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


    // -
WiFiClient wifiClient; // creating the instance for wificlient


PubSubClient client(server, 1883, callback ,wifiClient);

  //calling the predefined client id by passing parameter like server id,portand
  wificredential




void setup()// configureing the ESP32

{

    Serial.begin(115200); dht.begin(); pinMode(LED,OUTPUT); delay(10);
        Serial.println();

 wificonnect(); mqttconnect();

} void loop()// Recursive Function

    { h = dht.readHumidity(); t = dht.readTemperature(); Serial.print("temp:");
      Serial.println(t); Serial.print("Humid:"); Serial.println(h); PublishData(t, h);

delay(1000); if (!client.loop()) { mqttconnect();
```

```
    }
}

  void PublishData(float temp, float humid)

{

    mqttconnect();//function call for connecting to ibm
    /*  creating the String in in form JSon to update the data to ibm cloud */

  String payload = "{\"temp\":"; payload += temp; payload += "," "\"Humid\":";
payload += humid;                payload += "}";



 Serial.print("Sending payload: ");
 Serial.println(payload);



    if (client.publish(publishTopic, (char*) payload.c_str())) {

      Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print
  publish ok in Serial monitor or else it will print publish failed } else

{

     Serial.println("Publish failed");
    }




}


       void mqttconnect() {     if (!client.connected()) {
           Serial.print("Reconnecting client to ");
Serial.println(server);

       while (!!!client.connect(clientId, authMethod, token)) { Serial.print(".");
  delay(500);
       }
```

```
        initManagedDevice();
        Serial.println();

 }

 }
   void wificonnect() //function defination for wificonnect
{

    Serial.println(); Serial.print("Connecting to ");


  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
while (WiFi.status() != WL_CONNECTED) { delay(500);

      Serial.print(".");
    }
    Serial.println(""); Serial.println("WiFi connected"); Serial.println("IP address: ");
    Serial.println(WiFi.localIP());

}

void initManagedDevice() {

    if (client.subscribe(subscribetopic)) {
          Serial.println((subscribetopic)); Serial.println("subscribe to cmd OK");

    } else {

      Serial.println("subscribe to cmd FAILED");
}

}

  void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{



  Serial.print("callback invoked for topic: ");
Serial.println(subscribetopic);
```

```
    for (int i = 0; i < payloadLength; i++) {
  //Serial.print((char)payload[i]); data3+=
  (char)payload[i];
    }
    Serial.println("data: "+ data3); if(data3=="lighton")
    {
    Serial.println(data3); digitalWrite(LED,HIGH);
    }
    else
    {
Serial.println(data3); digitalWrite(LED,LOW);
} data3="";
}
```

Output:



8.Conclusion:

Through this project, we intend to obtain practical experience with the "Internet of Things" and "Embedded System" technologies that are now popular. IoT-enabled industrial monitoring systems are becoming more and more common in a wide range of industries because they raise safety standards by giving continuous alerts to workers and officials and real-time monitoring of vital factors like temperature, humidity, and smoke. The deployment could boost industry yields in addition to being done for safety concerns. The Internet of Things (IoT) is utilized in our project to gather data and facilitate online communication.In order to prevent accidents and hazards that are frequently ignored by industry people and users, we expect that our project will be useful enough to be implemented in industries across India. Utilizing the Industrial Internet of Things can help businesses in the industrial and logistics sectors better fulfill the demands of the new age of urgent needs (IoT).

9. REFERENCES

1. A Hazardous Area Personal Monitoring System for Operators in Gas Depots and Storage Tanks by Elia Landi, Lorenzo Parri, Ada Fort, Marco Mugnaini, Valerio Vignoli, Dinesh Tamang, Marco Tani

2. Embedded IoT-based Monitoring Utility for Safety Management and Access Control by Ugwechi WejieOkachi , Tamuno-Omie Joyce Alalibo and Emmanuel Chinweikpe Obuah

3. Monitoring of Hazardous Gases in Process Industries Through Internet by Ragavi P, Dr. K.R. Valluvan

4. IoT enabled environmental toxicology for air pollution monitoring using AI techniques by P Asha, LBTJRRGS Natrayan, BT Geetha, J Rene Beulah, R Sumathy, G Varalakshmi, S Neelakandan