

**Project Development Phase**  
**Sprint – 3 (SMS Alert Service & App Front – end Development)**

Date	11 November 2022
Team ID	PNT2022TMID35867
Project Name	Project – Gas leakage monitoring and alerting system for industries

**Software:**

- Arduino IDE
- MIT App Inventor

**Installed Libraries – Arduino IDE:**

- ESP8266WiFi
- LiquidCrystal\_I2C
- PubSubClient

**Source Code:**

```
#include <ESP8266WiFi.h>
#include <LiquidCrystal_I2C.h>
#include <PubSubClient.h>

#define Buzzer D5
#define Green D6
#define Sensor A0
#define ORG "wf2kmp"
#define DEVICE_TYPE "GLMASFI_IOT_Device_Cloud_Service"
#define DEVICE_ID "PNT2022TMID35867"
#define TOKEN "PNT2022TMID35867"

const char* ssid = "Airtel-Hotspot-958A";
const char* password = "9889i1bb";
const char* host = "maker.ifttt.com";
const int httpsPort = 80;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char topic[] = "iot-2/evt/status/fmt/json";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

void PublishData(float);
void callback(char*, byte*, unsigned int);

LiquidCrystal_I2C lcd(0x27, 16, 2);
WiFiClient client1;
WiFiClient client2;
PubSubClient client(server, 1883, callback, client2);

void setup() {
  lcd.backlight();
```

```

lcd.init();
pinMode(Green, OUTPUT);
pinMode(Buzzer, OUTPUT);
pinMode(Sensor, INPUT);
Serial.begin(115200);
Serial.println();

Serial.print("Connecting to ");
Serial.print(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.print("WiFi connected, IP address: ");
Serial.println(WiFi.localIP());
Serial.print("connecting to ");
Serial.println(host);
if (!client1.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return;
}
if (!client2.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return;
}
}

int msgSent = 0;
void loop() {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
        Serial.print(".");
        delay(500);
    }
    Serial.println();

    int sensor = analogRead(Sensor);
    Serial.println(String(sensor));
    PublishData(sensor);
    if (sensor >= 840 && !msgSent) {
        digitalWrite(Green, HIGH);
        digitalWrite(Buzzer, HIGH);
        String url = "/trigger/gasle/json/with/key/ktkqqpO7-nkuFo1Dc-jMZX4tNAKchaWS4E6SzY7btPA";
        Serial.print("Requesting URL: ");
        Serial.println(url);
        client1.print(String("GET ") + url + " HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection:
close\r\n\r\n");
        msgSent = 1;
        lcd.setCursor(0, 1);

```

```

    Serial.println("Gas Concentration Level: High");
}
else if (sensor >= 840 && msgSent) {
    digitalWrite(Green, HIGH);
    digitalWrite(Buzzer, HIGH);
    lcd.setCursor(0, 1);
    Serial.println("Gas Concentration Level: High");
}
else if (sensor >= 820) {
    digitalWrite(Buzzer, HIGH);
    digitalWrite(Green, HIGH);
    delay(750);
    digitalWrite(Buzzer, LOW);
    digitalWrite(Green, LOW);
    delay(1000);
    lcd.setCursor(0, 1);
    Serial.println("Gas Concentration Level: Moderate");
}
else {
    digitalWrite(Green, LOW);
    digitalWrite(Buzzer, LOW);
    lcd.setCursor(0, 1);
    Serial.println("Gas Concentration Level: Normal");
}
}
lcd.setCursor(0, 0);
lcd.print("Value: ");
lcd.print(sensor);
delay(1000);
}

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.println("callback invoked");
}

void PublishData(float senso){
    String payload;
    if(senso >= 840) {
        payload= "{\"Danger! High Gas Concentration\"":"";
    }
    else if(senso >= 820) {
        payload= "{\"Alert! Moderate Gas Concentration\"":"";
    }
    else {
        payload = "{\"Normal Gas Concentration\"":"";
    }
    payload += senso;
    payload += "}";

    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(topic, (char*) payload.c_str())) {
        Serial.println("Publish ok");
    }
}

```

```

else {
    Serial.println("Publish failed");
}
}
}

```

## Verification:

Concentration (ppm)	Level	IOT device Indication	Cloud Service Update	SMS Alert
> 840	High	Buzzer & LED - High	Danger! High Gas Concentration	Danger! Close gas Valve immediately – <b>(Only for Technician)</b> – Once
820 - 840	Moderate	Buzzer (with delay) & Blinking LED	Alert! Moderate Gas Concentration	NA
< 820	Normal	Buzzer & LED - Low	Normal Gas Concentration	NA

## Output:

### 1. Cloud Service:

Event	Value	Format	Last Received
status	{"Alert! Moderate Gas Concentration":822}	json	a few seconds ago
status	{"Alert! Moderate Gas Concentration":824}	json	a few seconds ago
status	{"Alert! Moderate Gas Concentration":838}	json	a few seconds ago
status	{"Danger! High Gas Concentration":878}	json	a few seconds ago
status	{"Danger! High Gas Concentration":867}	json	a few seconds ago

### 2. Serial Monitor Output:

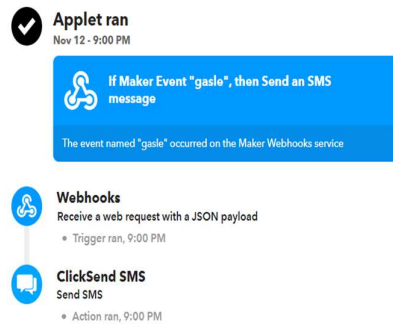
```

21:00:36.467 -> Connecting to Airtel-Hotspot-956A.....
21:00:42.690 -> Wifi connected, IP address: 192.168.1.100
21:00:42.690 -> connecting to maker.ifttt.com
21:00:42.865 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com
21:00:42.915 ->
21:00:43.857 -> 868
21:00:43.857 -> Sending payload: {"Danger! High Gas Concentration":868.00}
21:00:43.857 -> Publish ok
21:00:43.857 -> Requesting URL: /trigger/gasle/json/with/key/ktkqpg07-nkuFolDc-jMZX4tNAKchaWS4E6szI7btFA
21:00:43.857 -> Gas Concentration Level: High
21:00:44.841 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com
21:00:44.887 ->
21:00:44.887 -> 880
21:00:44.887 -> Sending payload: {"Danger! High Gas Concentration":880.00}
21:00:44.887 -> Publish ok
21:00:44.887 -> Gas Concentration Level: High
21:00:45.896 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com
21:00:45.896 ->
21:00:45.896 -> 867
21:00:45.896 -> Sending payload: {"Danger! High Gas Concentration":867.00}
21:00:45.896 -> Publish ok
21:00:45.896 -> Gas Concentration Level: High
21:00:46.896 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com
21:00:46.896 ->
21:00:46.896 -> 878
21:00:46.896 -> Sending payload: {"Danger! High Gas Concentration":878.00}
21:00:46.939 -> Publish ok
21:00:46.939 -> Gas Concentration Level: High
21:00:47.934 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com
21:00:47.934 ->
21:00:47.934 -> 838
21:00:47.934 -> Sending payload: {"Alert! Moderate Gas Concentration":838.00}
21:00:47.934 -> Publish ok
21:00:49.668 -> Gas Concentration Level: Moderate
21:00:50.686 -> Reconnecting client to wf2kmp.messaging.internetofthings.ibmcloud.com

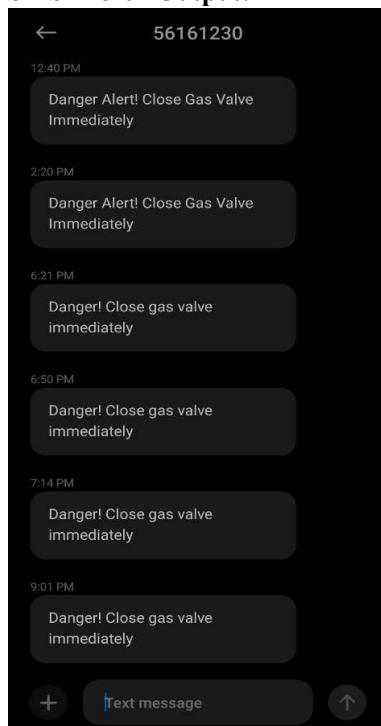
```

### 3. IFTTT Configuration & Request:

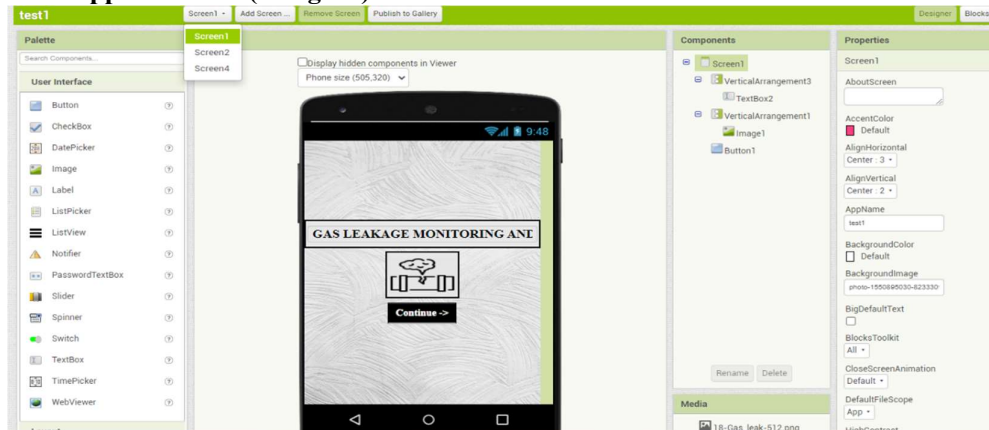
## If Maker Event "gasle", then Send an SMS message Activity

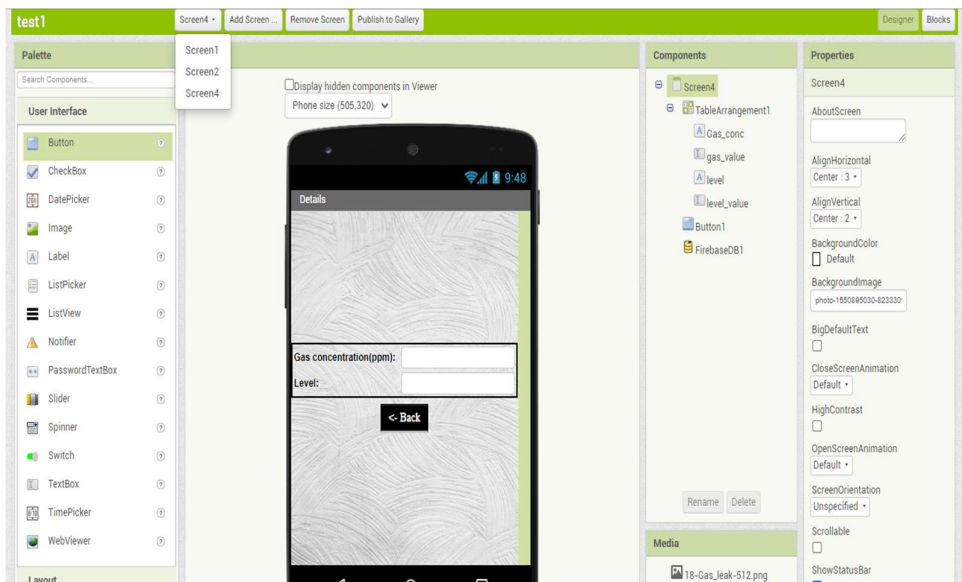
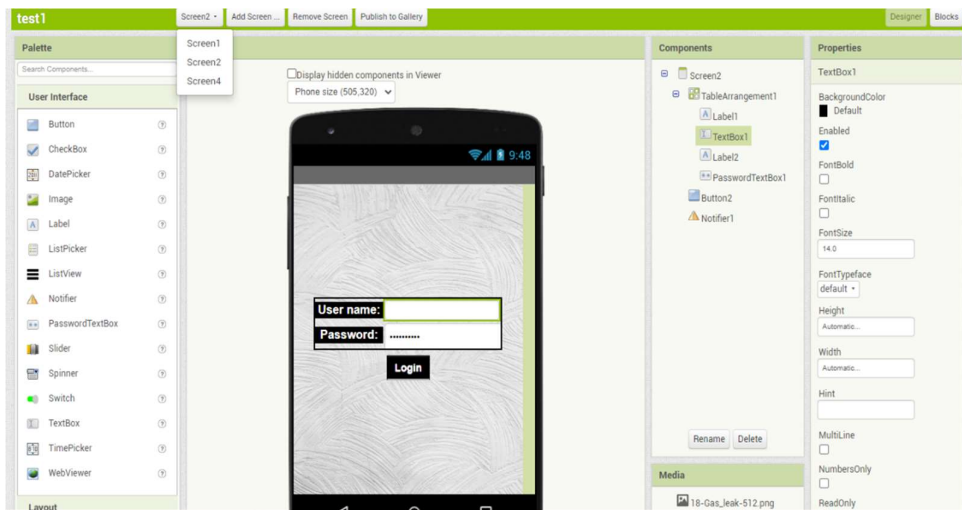


### 4. SMS Alert - Output:

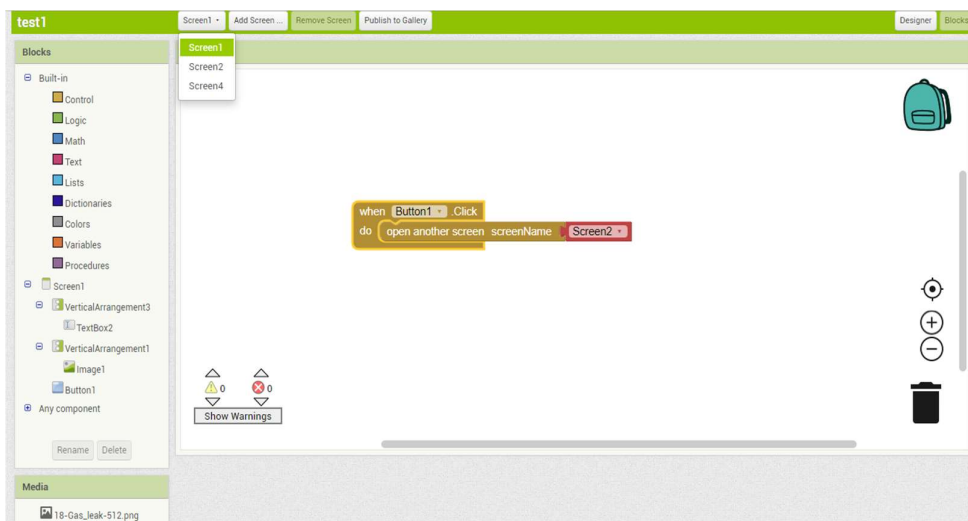


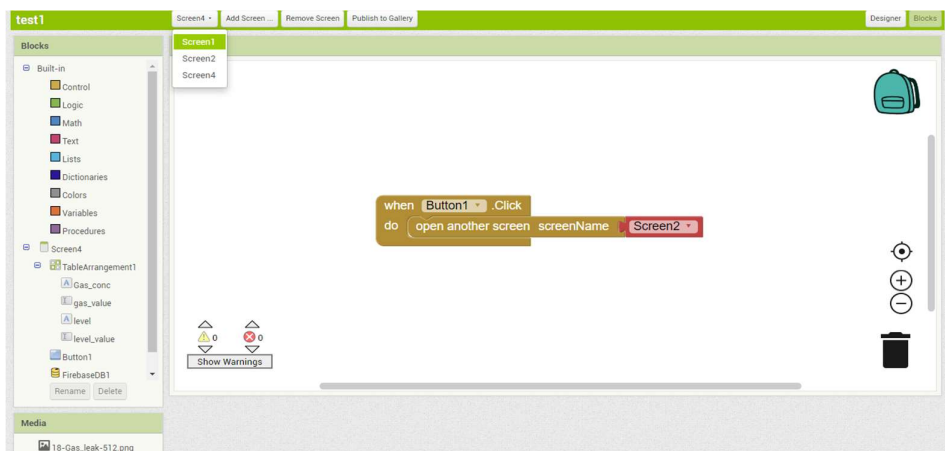
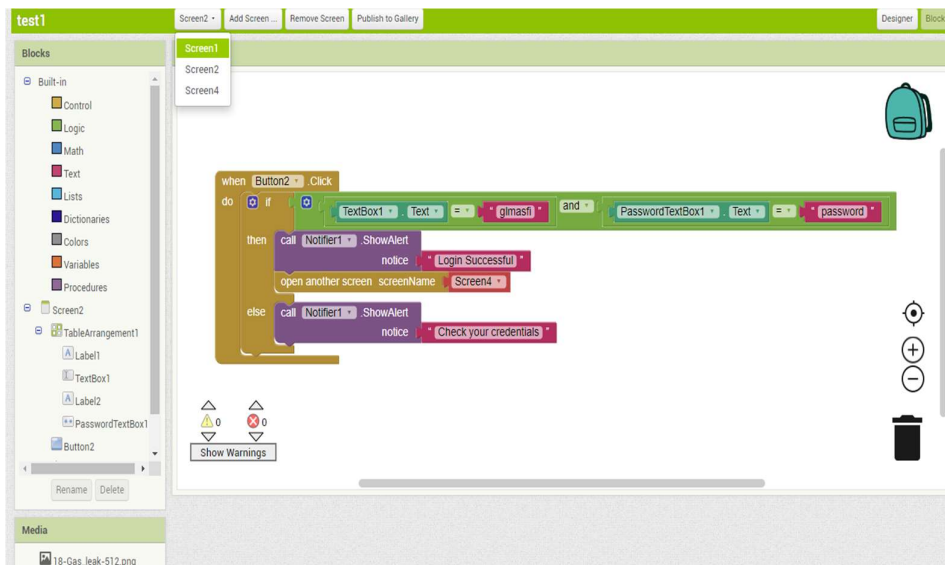
### MIT App-Inventor: (Designer)





## MIT App Inventor: (Blocks)





## Mobile App Output:

