correct codes:

TASK-1

Write an ARDUINO Program for Smart street light management system

---

```
void setup()
{
pinMode(A0,INPUT);
Serial.begin(9600);
pinMode(8,OUTPUT);
}
void loop()
{
int i=analogRead(A0);
Serial.println(i);
if(i>60)
digitalWrite(8,0);
else
digitalWrite(8,1);
delay(500);
}
```

TASK-2

Write an ARDUINO Program for Smart reserve break monitoring system

---

MANUAL CODE

```
#include <Ultrasonic.h>
Ultrasonic ultrasonic(11,12);//Init an Ultrasonic object
int Distance;
void setup() {
Serial.begin(9600);
}
void loop()
{
Distance=ultrasonic.Ranging(CM);//get the current result;
delay(100);
Serial.print("the distance is ");
Serial.println(Distance);
delay(1000);
}
```

TINKERCAD CODE

```
const int echoPin = 11; // Trigger pin
```

```
const int triggerPin= 12; // Echo pin

void setup() {

  Serial.begin(9600); // Start serial communication at 9600 bps

  pinMode(triggerPin, OUTPUT); // Set trigger pin as an OUTPUT

  pinMode(echoPin, INPUT); // Set echo pin as an INPUT

}

void loop() {

  long duration, distance;

  // Clear the trigger pin

  digitalWrite(triggerPin, LOW);

  delayMicroseconds(2);

  // Set the trigger pin HIGH for 10 microseconds

  digitalWrite(triggerPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(triggerPin, LOW);

  // Read the echo pin, duration is the time it takes for the pulse to return

  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance (duration/2) / 29.1 to convert to centimeters
  distance = (duration / 2) / 29.1;
  Serial.print("The distance is: ");
  Serial.print(distance);
  Serial.println(" cm");

  delay(100); // Wait for 1 second before the next reading
}
```

TASK-3

Write an ARDUINO Program for Smart soil state monitoring system

---

```
#include<SoftwareSerial.h>

void setup() {

// put your setup code here, to run once:

pinMode(A4,INPUT);

pinMode(13,OUTPUT);

Serial.begin(9600);

}

void loop() {

int sensorvalue=analogRead(A4);

Serial.println(sensorvalue);

if(sensorvalue<500)

{

Serial.println("led is on");

digitalWrite(13,HIGH);

delay(1000);

}
else
{
digitalWrite(13,LOW);
delay(1000);
}}
```

TASK-4

Write an ARDUINO Program for Micro controller-Activator Interface using DC Motor.

```
void setup(){
pinMode(10, OUTPUT);
pinMode(9, OUTPUT);
}
void loop(){
digitalWrite(10, LOW);
digitalWrite(9, HIGH);
delay(300);
digitalWrite(9,LOW);
digitalWrite(10, HIGH);
delay(300);
digitalWrite(9, LOW);
digitalWrite(10, LOW);
delay(300);
}
```

TASK-5

Write an ARDUINO Program for Moving curtail display using LCD

WRITEUP CODE

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(8,9,4,5,6,7);
int i=0;
char ch;
char s[]="Welcome to MC LAB!";
void setup()
{
lcd.begin(16,2);
}
void loop()
{
lcd.setCursor(0,0);
ch='!';
lcd.print(ch);
delay(1000);
lcd.clear();
lcd.setCursor(3,0);
for(i=0;s[i]!='\0';i++)
```

```
{

lcd.print(s[i]);

if(i==10)

lcd.setCursor(5,1);

}

delay(2000);

lcd.clear();}
```

++

EXECUTION CODE FOR TINKERCAD

```
#include <LiquidCrystal.h>

int seconds = 0;

LiquidCrystal lcd_1(12, 11, 5, 4, 3, 2);

void setup()
{
 lcd_1.begin(16, 2); // Set up the number of columns and rows on the LCD.
 // Print a message to the LCD.
 lcd_1.print("Welcome to MCIOT LAB");
}
void loop()
{
 // set the cursor to column 0, line 1

 // (note: line 1 is the second row, since counting

 // begins with 0):

 lcd_1.setCursor(0, 1);

 // print the number of seconds since reset:

 lcd_1.print(seconds);
```

```
 delay(1000); // Wait for 1000 millisecond(s)

 seconds += 1;

}
```

TASK-6

Write a GISMO Program for Sensor /Activator Interfacing using ESP32 using MPU 6050

MPU6050 (I2C Interface):

MPU6050 Pin ESP32 Pin

VCC 3.3V

GND GND

SDA GPIO 21

SCL GPIO 22

PIR Sensor:

PIR Sensor Pin ESP32 Pin

VCC 5V

GND GND

OUT (Signal) GPIO 26

WOKWI LINK

https://wokwi.com/projects/416800146063462401

```
#include "I2Cdev.h"

#include "MPU6050.h"

#include "Wire.h"

MPU6050 mpu;

float baseline[3];
```

```
float features[3];
float motion_threshold = 0.7;
void setup() {
  Wire.begin();
  Serial.begin(115200);
  mpu.initialize();
  calibrate();
  }
void loop() {
  float ax,ay,az;
  mpu_read(&ax,&ay,&az);
  ax=ax-baseline[0];
  ay=ay-baseline[1];
  az=az-baseline[2];
  mpu_record();
 delay(2000);
 }
void mpu_read(float *ax,float *ay,float *az) {
  int16_t _ax, _ay, _az, _gx, _gy, _gz;
  mpu.getMotion6(&_ax, &_ay, &_az, &_gx, &_gy, &_gz);
  *ax = _ax/16384.0;
  *ay = _ay/16384.0;
```

```
  *az = _az/16384.0;
}
void calibrate(){
 float ax,ay,az;
 for(int i=0;i < 10;i++){
  mpu_read(&ax,&ay,&az);
  delay(100);
   }
 baseline[0]=ax;
 baseline[1]=ay;
 baseline[2]=az;
}
void mpu_record(){
 float ax,ay,az;
 float aax,aay,aaz;
 String position;
  mpu_read(&ax,&ay,&az);
  ax = ax - baseline[0];
  ay = ay - baseline[1];
  az = az - baseline[2];
  features[0] = ax;
  features[1] = ay;
```

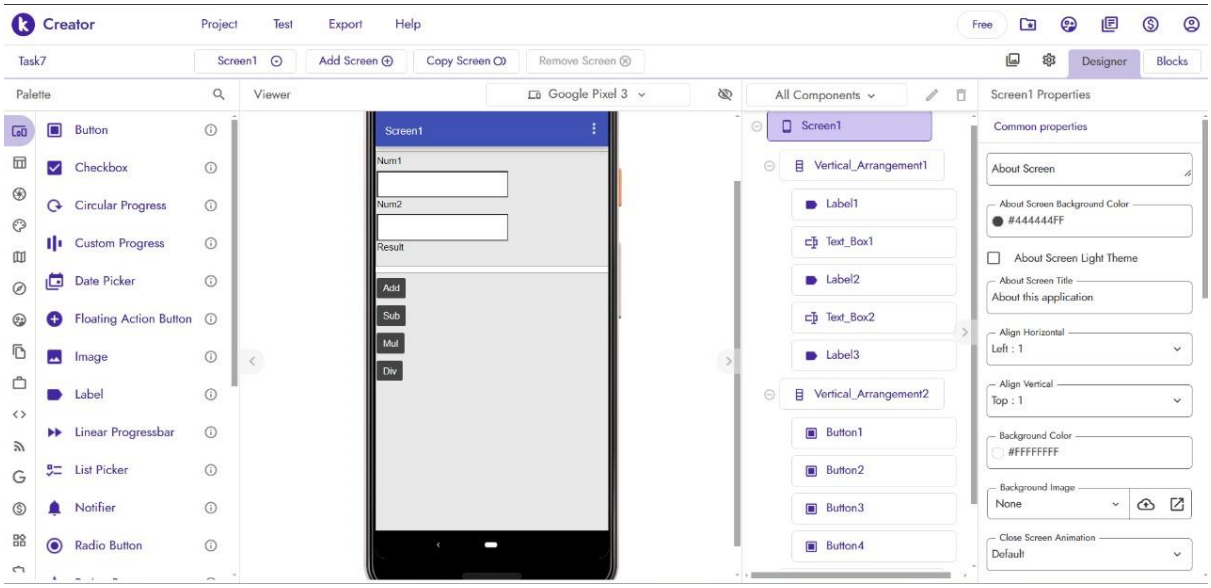
```
 features[2] = az;
Serial.print(features[0]);
Serial.print(" ");
Serial.print(features[1]);
Serial.print(" ");
Serial.println(features[2]);
aax=fabs(ax);
aay=fabs(ay);
aaz=fabs(az);
Serial.print(aax);
Serial.print(" ");
Serial.print(aay);
Serial.print(" ");
Serial.println(aaz);
position="Upright";
if(aax > motion_threshold)
{
 if(ax > 0)
 position="Left";
 else
 position="Right";
}
```

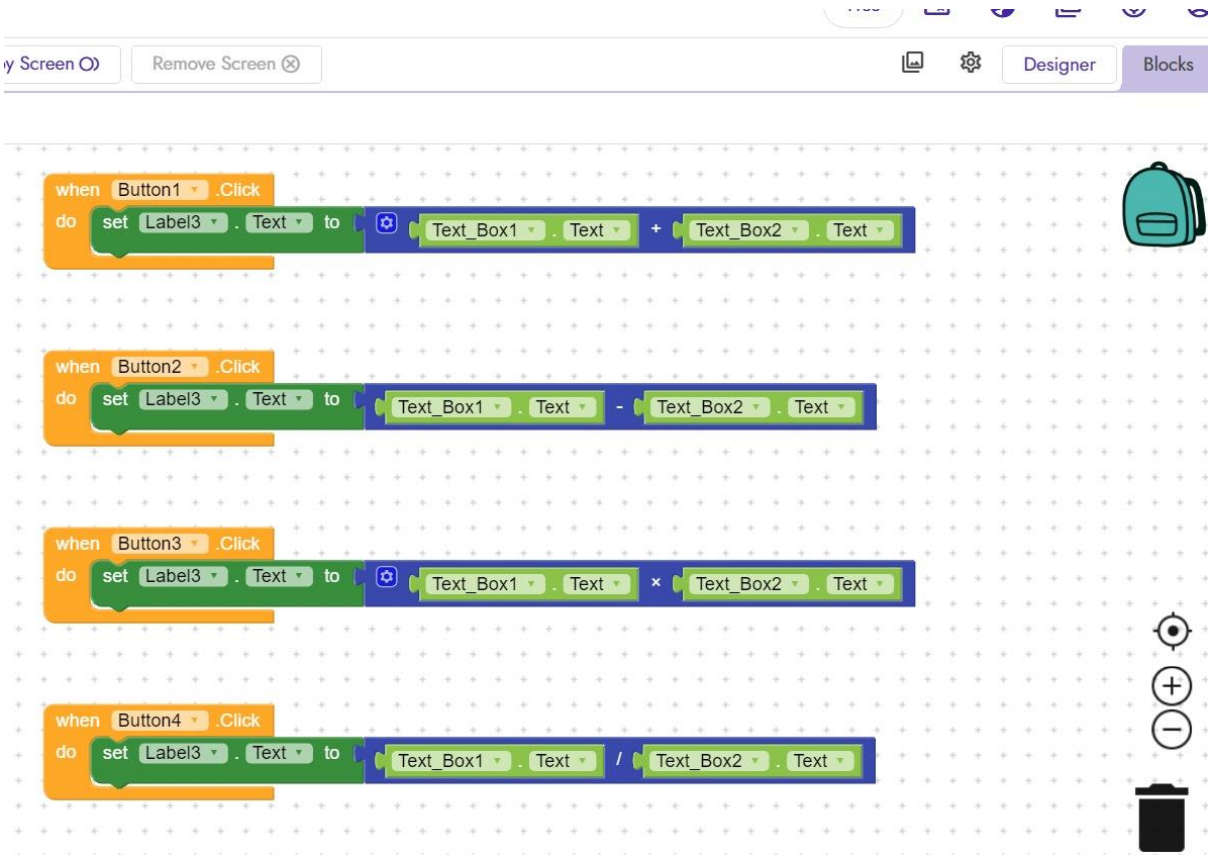
```
if(aay > motion_threshold)

{

 if(ay > 0)

 position="Backward";

 else
 position="Forward";
}
Serial.println(position);
}
```

TASK-7A,7B,7C

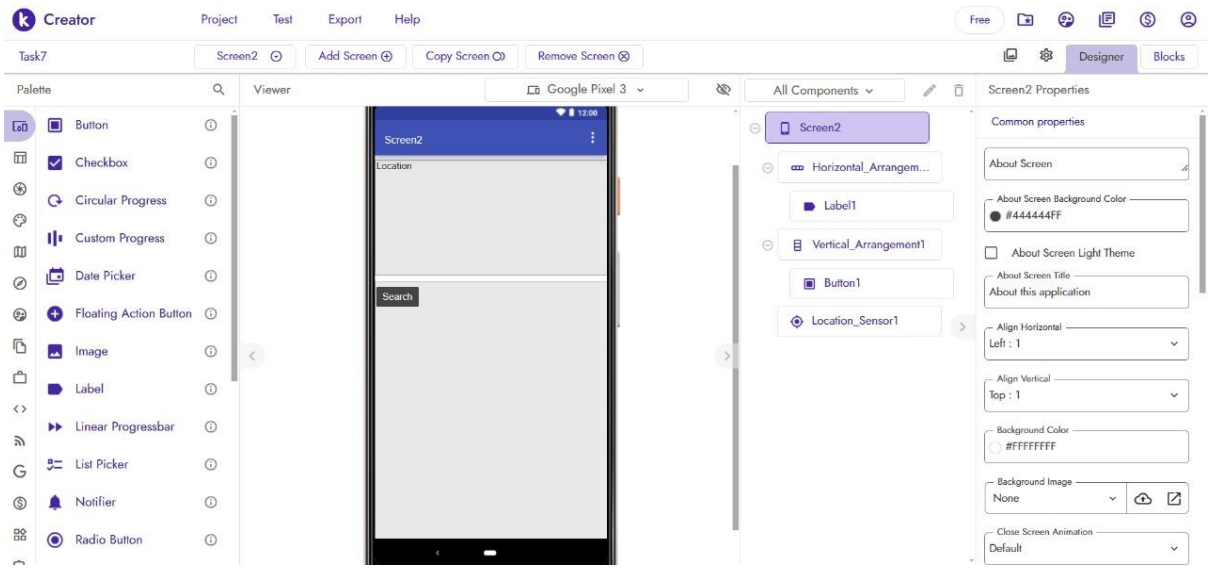
Write a Program to develop Mobile App

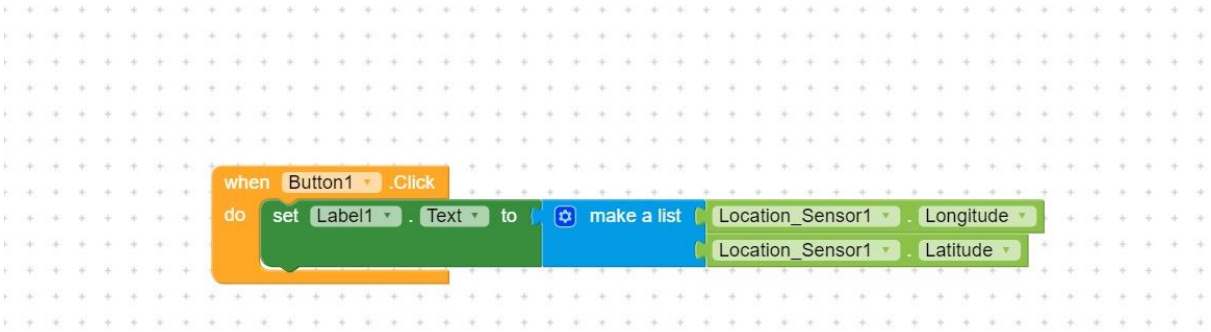
a) Simple interface (Calculator)



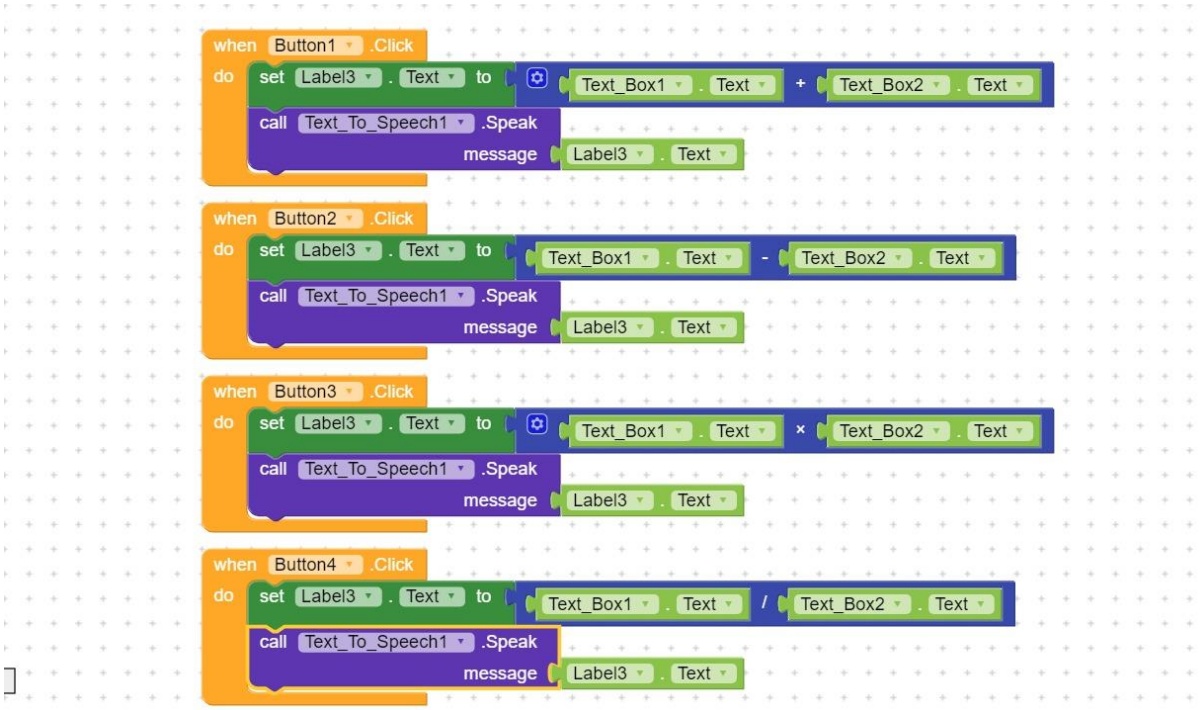


b) Location Sensor





c)Text to speech Converter



TASK-8

Write a program using GISMO VI board for Internet enabled remote range indicator

---

```
#include "Credentials.h"

// defines pins numbers
const int trigPin = 25;
const int echoPin = 26;

// defines variables
long duration;
int distance;

void OLEDInit();
void OLEDUpdate();
void WiFiInit();
void FirebaseInit();
void FirebaseWrite();

String tag = "IOTLAB/Range_Meter/Range";


void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(115200); // Starts the serial communication
OLEDInit();
WiFiInit();
FirebaseInit();
}

void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
```

```
digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);

// Calculating the distance
distance= duration*0.034/2;

// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);

OLEDUpdate();
FirebaseWrite();

delay(1000);
}
void OLEDInit(){
  display.init();
  display.setFont(ArialMT_Plain_24);
}

void OLEDUpdate(){

  String d = String(distance) + " cm";
  display.clear();
  display.drawString(30,0,d);
  display.display();
}
void WiFiInit(){
  pinMode(2,OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    digitalWrite(2,!digitalRead(2));
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
```

```
}
void FirebaseInit(){
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);

}
void FirebaseWrite(){
  Firebase.setInt(firebaseData,tag,distance);

}
```

TASK-9

Write a program using GISMO VI board for Internet enabled smart room lighting system

---

Relay Sensor ESP32 pins

NC/NO with COM GPIO13

GND GND

Vcc 3.3V

```
void setup()
// put your setup code here, to run once:
pinMode(13,OUTPUT);
}
void loop() {
// put your main code here, to run repeatedly:
digitalWrite(13,HIGH);
delay(2000);
digitalWrite(13,LOW);
delay(2000);
}
```

TASK-10

Write a program using GISMO VI board for Internet enabled Smart Garden Maintenance

---

```
/* Soil Moisture sensor pin   : GPIO34
 * Relay pin                 : GPIO13
 *
  */
#include "Credentials.h"

#define smPin 34
#define relayPin 13
int smValue;
int smLimit = 1500;
int delayTime = 2000;
String motorStatus = "OFF";
void OLEDInit();
void OLEDUpdate();
void WiFiInit();
void FirebaseInit();
void FirebaseWrite();
void FirebaseRead();

String tag1 = "IOTLAB/Smart_Garden/Soil_Moisture";
String tag2 = "IOTLAB/Smart_Garden/Motor_Status";


void setup() {
  // put your setup code here, to run once:
  pinMode(relayPin,OUTPUT);
  Serial.begin(115200);
  OLEDInit();
  WiFiInit();
  FirebaseInit();
  }

void loop() {
  // put your main code here, to run repeatedly:

FirebaseRead();
smValue = 4095 - analogRead(smPin);
Serial.print("Soil Moisture = ");
Serial.println(smValue);
```

```
if (smValue < smLimit){
digitalWrite(relayPin,HIGH);
Serial.println("Motor turned ON");
motorStatus = "ON";
}
else{
digitalWrite(relayPin,LOW);
Serial.println("Motor turned OFF");
motorStatus = "OFF";
}

OLEDUpdate();
FirebaseWrite();
delay(delayTime);
}

void OLEDInit(){
  display.init();
  display.setFont(ArialMT_Plain_24);
}

void OLEDUpdate(){


  display.clear();
  display.drawString(30,0,String(smValue));
  display.drawString(30,30,String(motorStatus));
  display.display();
}

void WiFiInit(){
  pinMode(2,OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    digitalWrite(2,!digitalRead(2));
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
```

```
  Serial.println();

}
void FirebaseInit(){
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);

}
void FirebaseWrite(){
  Firebase.setInt(firebaseData,tag1,smValue);
  Firebase.setString(firebaseData,tag2,motorStatus);

}

void FirebaseRead(){
String smCloudFull;
String smCloud;
if(Firebase.getString(firebaseData,"IOTLAB/Smart_Garden/SM_Threshold",smCloudFull)){
smCloud = smCloudFull.substring(2,smCloudFull.length()-2);
smLimit = smCloud.toInt();
Serial.println(smLimit);
}
}
```

TASK-11

Internet enabled Display of Ambient Prameter(BMP).

```
#include <Wire.h>
#include <Adafruit_BMP280.h>


Adafruit_BMP280 bmp; // I2C
#include "Credentials.h"

#define SEALEVELPRESSURE_HPA (1013.25)

float tempC,tempF,atPressure,altitude,humidity;
void setup() {
  Serial.begin(115200);
  Serial.println(F("BMP280 test"));

  if (!bmp.begin(BMP280_ADDRESS_ALT, BMP280_CHIPID)) {
     Serial.println(F("Could not find a valid BMP280 sensor, check wiring or "
                      "try a different address!"));
    while (1) delay(10);
  }

  /* Default settings from datasheet. */
  bmp.setSampling(Adafruit_BMP280::MODE_NORMAL,     /* Operating Mode. */
                  Adafruit_BMP280::SAMPLING_X2,     /* Temp. oversampling */
                  Adafruit_BMP280::SAMPLING_X16,    /* Pressure oversampling */
                  Adafruit_BMP280::FILTER_X16,      /* Filtering. */
                  Adafruit_BMP280::STANDBY_MS_500); /* Standby time. */

 OLEDInit();
 WiFiInit();
 FirebaseInit();
}

void loop() {
   printValues();
   displayValues();
   delay(2000);
}

 void OLEDInit(){
```

```
  display.init();
  display.setFont(ArialMT_Plain_16);
}

void printValues(){
 // float tempC,tempF,atPressure,altitude,humidity;
  Serial.print("Temperature = ");
  tempC = bmp.readTemperature();
//Serial.print(bme.readTemperature());
Serial.print(tempC);
Serial.println(" deg C");
Firebase.setFloat(firebaseData,"IOTLAB/Environment_Monitor/Temperature",tempC);

Serial.print("Temperature = ");
//Serial.print(1.8*bme.readTemperature()+32);
Serial.print(1.8*tempC+32);
Serial.println(" deg F");

Serial.print("Pressure = ");
atPressure=bmp.readPressure()/100.0F;
//Serial.print(bme.readPressure()/100.0F);
Serial.print(atPressure);
Serial.println(" hPa");
Firebase.setFloat(firebaseData,"IOTLAB/Environment_Monitor/Pressure",atPressure);

Serial.print("Altitude = ");
altitude=bmp.readAltitude(SEALEVELPRESSURE_HPA);
//Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
Serial.print(altitude);
Serial.println(" m");
Firebase.setFloat(firebaseData,"IOTLAB/Environment_Monitor/Altitude",altitude);
Serial.println();
Firebase.setFloat(firebaseData,"IOTLAB/Environment_Monitor/Humidity",0.0);

}

void displayValues(){
  display.clear();

  String myString = "";

  char buffer[6];
  dtostrf(tempC,5,1,buffer);
  myString.concat(buffer);
```

```
  myString.concat(" C");
  display.drawString(0,0,myString);

  humidity = 0.0;
  myString = "";
  dtostrf(humidity,5,1,buffer);
  myString.concat(buffer);
  myString.concat(" %");
  display.drawString(64,0,myString);

 myString = "";
  dtostrf(atPressure,5,1,buffer);
  myString.concat(buffer);
   display.drawString(0,30,myString);

    myString = "";
  dtostrf(altitude,5,1,buffer);
  myString.concat(buffer);
  myString.concat("m");
   display.drawString(64,30,myString);


  display.display();
}
void WiFiInit(){
  pinMode(2,OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    digitalWrite(2,!digitalRead(2));
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
}
void FirebaseInit(){
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);}
```

TASK-12

Internet Enabled Home Safety and security system.

```
#define magSwitch 16
#define PIR 33
#define LED 2



#include "Credentials.h"

String magswStatus;
String PIRStatus;

int loopDelay = 1000;

void OLEDInit();
void OLEDUpdate();
void WiFiinit();
void FirebaseInit();
void FirebaseWrite();

String tag1 = "IOTLAB/HSS/MagswStatus";
String tag2 = "IOTLAB/HSS/PIRSwitch";
void setup() {
  // put your setup code here, to run once:
  pinMode(magSwitch,INPUT_PULLUP);
  pinMode(PIR,INPUT);
```

```
  pinMode(LED,OUTPUT);
  Serial.begin(115200);
  OLEDInit();
  WiFiInit();
  FirebaseInit();

  // Initial warmup period of 1 minute for PIR sensor
  for(int i = 0; i < 60; i++){
    digitalWrite(2,!digitalRead(2));
    delay(1000);
  }
}

void loop() {
  // put your main code here, to run repeatedly:
// Mag switch reading
 int sw_status = digitalRead(16);
 if(sw_status == HIGH){
 Serial.println("OPEN");
 magswStatus = "Window_Open";}
 else{
 Serial.println("CLOSED");
 magswStatus = "Window_Closed";}

// PIR sensor reading
  int pir_status = digitalRead(PIR);
  if (pir_status == HIGH) {
    Serial.println("Motion detected!");
    PIRStatus = "Motion_detected";
```

```
    loopDelay = 3000;
    }
  else {
    Serial.println("No Motion");
    PIRStatus = "No_Motion";
    loopDelay = 1000;
    }
OLEDUpdate();
FirebaseWrite();
 delay(loopDelay);
}
void OLEDInit(){
  display.init();
  display.setFont(ArialMT_Plain_16);
  display.clear();
  display.drawString(0,0,"Warming up...");
  display.display();
}

void OLEDUpdate(){
  display.clear();
  display.drawString(0,0,magswStatus);
  display.drawString(0,30,PIRStatus);
  display.display();
}
void WiFiInit(){
  pinMode(2,OUTPUT);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
```

```
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    digitalWrite(2,!digitalRead(2));
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

}
void FirebaseInit(){
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);

}
void FirebaseWrite(){
  Firebase.setString(firebaseData,tag1,magswStatus);
  Firebase.setString(firebaseData,tag2,PIRStatus);

}
```
