Code

# Properties

#include <ArduinoIoTCloud.h>

#include <Arduino\_ConnectionHandler.h>

const char DEVICE\_LOGIN\_NAME[] = "17ad7fc0-343a-43e4-8317-9166130a5a01";

const char SSID[] = "Smart Ext";                      // Network SSID (name)

const char PASS[] = "SmartExt";                    // Network password (use for WPA, or use as key for WEP)

const char DEVICE\_KEY[] = "LVGELQMVDAOEPZOT1QKP";  // Secret device password

int localStatus;

void onStatusChange();

void onPowerChange();

void onTestingChange();

int status = 3;

float smoke;

bool power = true;

bool testing = false;

CloudTemperature temp;

void initProperties() {

  ArduinoCloud.setBoardId(DEVICE\_LOGIN\_NAME);

  ArduinoCloud.setSecretDeviceKey(DEVICE\_KEY);

  ArduinoCloud.addProperty(status, READWRITE, ON\_CHANGE, onStatusChange);

  ArduinoCloud.addProperty(power, READWRITE, ON\_CHANGE, onPowerChange);

  ArduinoCloud.addProperty(testing, READWRITE, ON\_CHANGE, onTestingChange);

  ArduinoCloud.addProperty(temp, READ, 10 \* SECONDS, NULL);

  ArduinoCloud.addProperty(smoke, READ, 10 \* SECONDS, NULL);

}

WiFiConnectionHandler ArduinoIoTPreferredConnection(SSID, PASS);

# Main

#include "thingProperties.h"

void setup() {

  Serial.begin(19200);

  delay(1500);

  initStatus();

  initProperties();

  setupTemp();

  initLcd();

  initPower();

  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  setDebugMessageLevel(2);

  ArduinoCloud.printDebugInfo();

}

void loop() {

  ArduinoCloud.update();

  tempLoop();

  smkLoop();

  statusLoop();

  powerLoop();

  showStatus();

  displayLoop();

}

void onStatusChange() {

  if (testing) localStatus = status;

}

void onPowerChange() {}

void onTestingChange() {}

# LCD Display

#include <LiquidCrystal\_I2C.h>

#define BLANK "                "

#define ON "ON"

#define OFF "OFF"

#define ONLINE "Online"

#define OFFLINE "Offline"

#define WARN "Warning"

#define ERR "Error"

LiquidCrystal\_I2C lcd(0x27, 16, 2);

String prevPower;

String prevStatus;

void initLcd() {

  lcd.init();

  lcd.clear();

  lcd.backlight();

  print("Smart Extension");

  delay(1000);

  print("Power : ");

  print2("Status: ");

}

void print(const char\* text) {

  lcd.setCursor(0, 0);

  lcd.print(BLANK);

  lcd.setCursor(0, 0);

  lcd.print(text);

}

void print2(const char\* text) {

  lcd.setCursor(0, 1);

  lcd.print(BLANK);

  lcd.setCursor(0, 1);

  lcd.print(text);

}

void printPower() {

  String s;

  switch (localStatus) {

    case 1:

      {

        s = WARN;

        break;

      }

    case 2:

      {

        s = ERR;

        break;

      }

    case 3:

      {

        s = OFFLINE;

        break;

      }

    case 0:

    default:

      {

        s = ONLINE;

      }

  }

  if (prevStatus != s) {

    lcd.setCursor(8, 1);

    lcd.print("         ");

    lcd.setCursor(8, 1);

    lcd.print(s.c\_str());

    prevStatus = s;

  }

}

void printStatus() {

  String currPower = power ? ON : OFF;

  if (prevPower != currPower) {

    lcd.setCursor(8, 0);

    lcd.print("         ");

    lcd.setCursor(8, 0);

    lcd.print(currPower);

    prevPower = currPower;

  }

}

  void displayLoop() {

    printPower();

    printStatus();

  }

# Power

#define POWER D0

void initPower() {

  pinMode(POWER, OUTPUT);

  digitalWrite(POWER, LOW);

}

void powerLoop() {

  power = power && localStatus != 2;

  if (power) {

    digitalWrite(POWER, LOW);

  } else {

    digitalWrite(POWER, HIGH);

  }

}

# Smoke Sensor

int mq2 = A0;

void smkLoop() {

  smoke = analogRead(mq2);

}

# LED Status

int mq2 = A0;

void smkLoop() {

  smoke = analogRead(mq2);

}

# Temperature Sensor

#include <OneWire.h>

#include <DallasTemperature.h>

// Data wire is connected to GPIO2

#define ONE\_WIRE\_BUS 2

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

void setupTemp() {

  sensors.begin();

}

void tempLoop() {

  sensors.requestTemperatures();

  temp = sensors.getTempCByIndex(0);

}