FOOD PERISH PREVENTION SYSTEM USING IOT

ARDUINO CODE:

```
#include <DHT.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include <Wire.h>

#include <LiquidCrystal_I2C.h>


#define DHTPIN 2

#define ONE_WIRE_BUS 3

#define LM35PIN A0

#define RELAY_HEAT 5

#define RELAY_COOL 4


#define BTN_HOT_MODE 6

#define BTN_COLD_MODE 7

#define BTN_SANDWICH 8

#define BTN_PUFFS 9

#define BTN_MILK 12

#define BTN_DRINKS 13


#define DHTTYPE DHT11


DHT dht(DHTPIN, DHTTYPE);

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);

LiquidCrystal_I2C lcd(0x27, 16, 2);


bool modeSelected = false;

bool foodSelected = false;
bool isColdMode = false;
String foodType = "";

float tempMin = 0.0;
float tempMax = 0.0;

void printLCDAndSerial(int row, const char* msg) {
 lcd.setCursor(0, row);
 lcd.print("                ");
 lcd.setCursor(0, row);
 lcd.print(msg);
 Serial.println(msg);
}

void blinkAlert() {
 while (true) {
  lcd.clear();
  lcd.setCursor(3, 0);
  lcd.print("!!! ALERT !!!");
  lcd.setCursor(3, 1);
  lcd.print("Sensor Error");
  Serial.println("!!! ALERT: Sensor error detected !!!");

  digitalWrite(RELAY_COOL, HIGH);
  digitalWrite(RELAY_HEAT, HIGH);
  delay(700);
  lcd.clear();
  delay(300);
 }
}

void setup() {
 Serial.begin(115200);
 dht.begin();
 sensors.begin();
 lcd.init();
 lcd.backlight();

 pinMode(RELAY_COOL, OUTPUT);
 pinMode(RELAY_HEAT, OUTPUT);
 digitalWrite(RELAY_COOL, HIGH);
 digitalWrite(RELAY_HEAT, HIGH);

 pinMode(BTN_HOT_MODE, INPUT_PULLUP);
 pinMode(BTN_COLD_MODE, INPUT_PULLUP);
 pinMode(BTN_SANDWICH, INPUT_PULLUP);
 pinMode(BTN_PUFFS, INPUT_PULLUP);
 pinMode(BTN_MILK, INPUT_PULLUP);
 pinMode(BTN_DRINKS, INPUT_PULLUP);

 lcd.clear();
 printLCDAndSerial(0, "Select Mode:");
 printLCDAndSerial(1, "HOT / COLD");
}

void loop() {
 if (!modeSelected) {
  if (digitalRead(BTN_HOT_MODE) == LOW) {
   modeSelected = true;
   isColdMode = false;
   lcd.clear();
   printLCDAndSerial(0, "HOT Mode Selected");
   delay(1000);
   lcd.clear();
   printLCDAndSerial(0, "Select Food:");
   printLCDAndSerial(1, "Puffs / Sand.");
   Serial.println("Mode: HOT selected");
  }
  else if (digitalRead(BTN_COLD_MODE) == LOW) {
   modeSelected = true;
   isColdMode = true;
   lcd.clear();
   printLCDAndSerial(0, "COLD Mode Selected");
   delay(1000);
   lcd.clear();
   printLCDAndSerial(0, "Select Food:");
   printLCDAndSerial(1, "Milk / Drinks");
   Serial.println("Mode: COLD selected");
  }
  return;
 }

 if (!foodSelected) {
  if (isColdMode) {
   if (digitalRead(BTN_MILK) == LOW) {
    foodType = "Milk";
    tempMin = 1.0;
    tempMax = 4.0;
    foodSelected = true;
   }
   else if (digitalRead(BTN_DRINKS) == LOW) {
    foodType = "Drinks";
    tempMin = 5.0;
    tempMax = 8.0;
    foodSelected = true;
   }
 } else {
  if (digitalRead(BTN_PUFFS) == LOW) {
   foodType = "Puffs";
   tempMin = 60.0;
   tempMax = 70.0;
   foodSelected = true;
  }
  else if (digitalRead(BTN_SANDWICH) == LOW) {
   foodType = "Sandwich";
   tempMin = 50.0;
   tempMax = 60.0;
   foodSelected = true;
  }
 }

 if (foodSelected) {
  lcd.clear();
  printLCDAndSerial(0, "Food Selected:");
  printLCDAndSerial(1, foodType.c_str());
  delay(1500);
  lcd.clear();
 }
 return;
}

float lm35Temp = analogRead(LM35PIN) * (5.0 / 1023.0) * 100.0;
float dhtTemp = dht.readTemperature();
float dhtHum = dht.readHumidity();
sensors.requestTemperatures();
float ds18Temp = sensors.getTempCByIndex(0);

// Alert on sensor error
```

```
  if (isnan(dhtHum) || isnan(dhtTemp) || ds18Temp == -127.0 || lm35Temp < -20 || lm35Temp >
100) {
    blinkAlert();
  }

  bool coolOn = false, heatOn = false;
  if (ds18Temp > tempMax && isColdMode) {
    digitalWrite(RELAY_COOL, LOW);
    digitalWrite(RELAY_HEAT, HIGH);
    coolOn = true;
  }
  else if (ds18Temp < tempMin && !isColdMode) {
    digitalWrite(RELAY_COOL, HIGH);
    digitalWrite(RELAY_HEAT, LOW);
    heatOn = true;
  }
  else {
    digitalWrite(RELAY_COOL, HIGH);
    digitalWrite(RELAY_HEAT, HIGH);
  }

  // Serial debug output
  Serial.println("=== Sensor Readings ===");
  Serial.print("LM35 Temp: "); Serial.println(lm35Temp);
  Serial.print("DHT11 Temp: "); Serial.println(dhtTemp);
  Serial.print("Humidity: "); Serial.println(dhtHum);
  Serial.print("DS18B20 Temp: "); Serial.println(ds18Temp);
  Serial.print("Cooling Relay: "); Serial.println(coolOn ? "ON" : "OFF");
  Serial.print("Heating Relay: "); Serial.println(heatOn ? "ON" : "OFF");
  Serial.print("Selected Food: "); Serial.println(foodType);
  Serial.println("======================");

  // LCD Output
```

```
 lcd.setCursor(0, 0);
 lcd.print("T:");
 lcd.print(ds18Temp, 1);
 lcd.print((char)223);
 lcd.print("C ");

 if (coolOn) lcd.print("C:ON ");
 else if (heatOn) lcd.print("H:ON ");
 else lcd.print("STBY  ");

 char line2[17];
 snprintf(line2, sizeof(line2), "H:%d%% %s", (int)dhtHum, foodType.c_str());
 lcd.setCursor(0, 1);
 lcd.print("                ");
 lcd.setCursor(0, 1);
 lcd.print(line2);

 delay(5000);
}
```

ESP01 Wifi Module Code

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>

const char* ssid = "<Wifi_Name>";
const char* password = "<Wifi_Password>";
const char* host = "api.thingspeak.com";
const char* apiKey = "5WD8MGBI66TQXEHT";

WiFiClient client;
```

```
void setup() {
 Serial.begin(9600); // Connect to Arduino
 delay(1000);

 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.println("Connecting to WiFi...");
 }

 Serial.println("Connected to WiFi!");
 Serial.print("IP Address: ");
 Serial.println(WiFi.localIP());
}

void loop() {
 if (Serial.available()) {
  String data = Serial.readStringUntil('\n');
  data.trim(); // remove whitespace or \r
  Serial.println("Received from Arduino: " + data);

  // Split the data
  int idx1 = data.indexOf(',');
  int idx2 = data.indexOf(',', idx1 + 1);
  int idx3 = data.indexOf(',', idx2 + 1);

  if (idx1 > 0 && idx2 > idx1 && idx3 > idx2) {
   String val1 = data.substring(0, idx1);           // ds18Temp
   String val2 = data.substring(idx1 + 1, idx2);     // dhtTemp
   String val3 = data.substring(idx2 + 1, idx3);     // lm35Temp
   String val4 = data.substring(idx3 + 1);           // dhtHum
```

```
      if (client.connect(host, 80)) {
        String url = "/update?api_key=" + String(apiKey) +
                     "&field1=" + val1 +
                     "&field2=" + val2 +
                     "&field3=" + val3 +
                     "&field4=" + val4;

        client.print(String("GET ") + url + " HTTP/1.1\r\n" +
                     "Host: " + host + "\r\n" +
                     "Connection: close\r\n\r\n");

        Serial.println("Data sent to ThingSpeak!");
        Serial.println("URL: " + url);
      } else {
        Serial.println("Connection to ThingSpeak failed.");
      }
    } else {
      Serial.println("Invalid data format received.");
    }

    delay(2000); // optional delay between uploads
  }
}
```