# پروژه خانه ی هوشمند

## معرفی پروژه

تاکنون در این کتاب روش استفاده از انواع سنسور ها در کنار برد های Arduino آموزش داده شد. حال با قرار دادن تمامی سنسور ها در کنار هم یک خانه ی هوشمند هر چند ساده اما کاربردی تولید می­کنیم.

این پروژه از دو بخش تشکیل می­شود:

1. گلخانه ی هوشمند
2. امنیت خانه

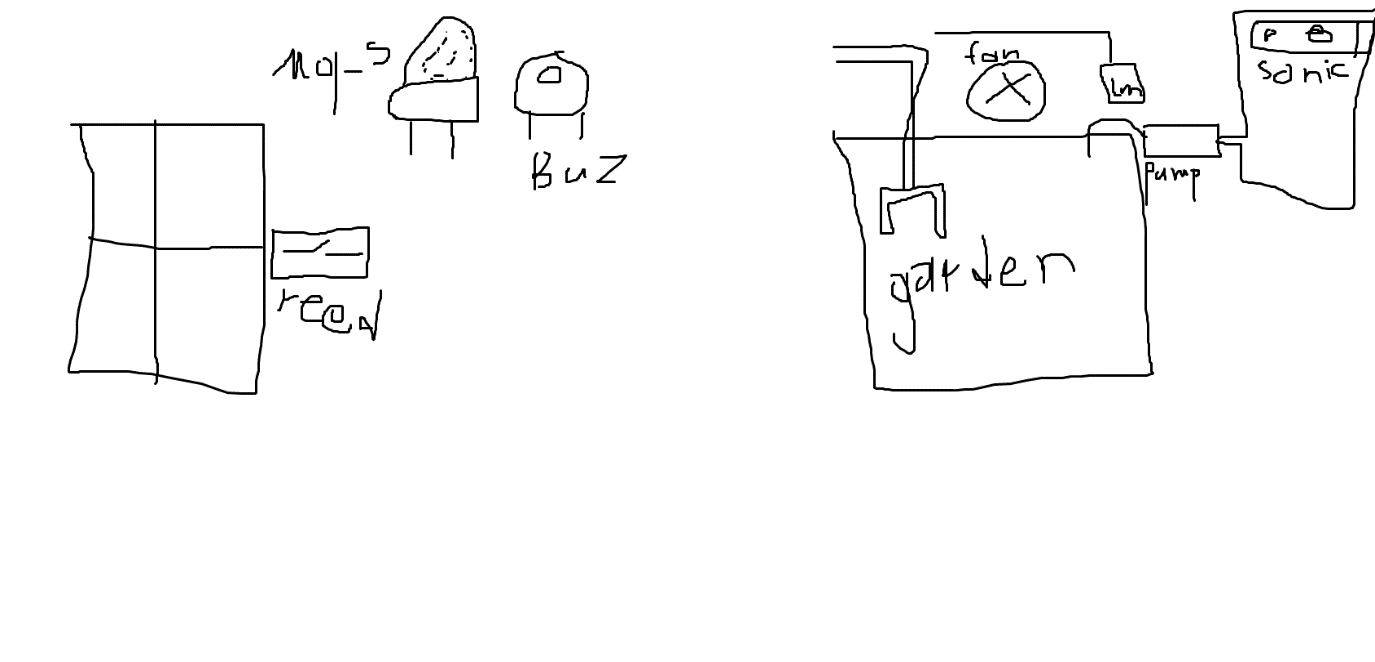
در قسمت گلخانه ی هوشمند، سعی می­شود یک گلخانه ی هوشمند که توانایی کنترل دما و رطوبت محیط و خاک را داشته باشد تولید شود. و در قسمت امنیت خانه کیفیت هوا بررسی شده و گزارش می­شود همچنین بر روی پنجره اتاق دزدگیر نصب می­شود.

### گلخانه هوشمند

در این گلخانه دما و رطوبت به صورت هوشمند کنترل می­شود. بدین صورت که اگر دما از حد مشخصی بیشتر شد فن شروع به کار میکند و با تهویه ی هوا دما را کاهش می­دهد همچنین اگر رطوبت خاک کم شد پمپ آب فعال شده و خاک را تر می­کند. پمپ آب به یک مخزن کوچک آب متصل است که به صورت دستی پر می­شود. میزان آب موجود در مخزن اندازه گیری و گزارش می-شود.

در این گلخانه از سنسور YL-69 جهت تشخیص رطوبت خاک ، از سنسور DS18B20جهت اندازه گیری دمای هوا و از سنسور فراصوت HC-SR05 جهت اندازه گیری مقدار آب موجود در مخزن استفاده می­شود. همچنین از یک پمپ آب جهت تنظیم رطوبت خاک و از یک فن کوچک جهت کنترل دمای هوا استفاده می­شود. یک نمایشگر LCD16\*2 به همراه تبدیل I2C جهت نمایش اطلاعات مورد استفاده قرار می­گیرد.

### امنیت خانه

در زمینه ی امنیت خانه از سنسور reed switch جهت تولید دزدگیر، از سنسور PIR جهت روشن کردن چراغ و از سنسور MQ-5 جهت اندازه گیری کیفیت هوا استفاده می­شود همچنین از یک buzzer جهت اطلاع رسانی نامناسب بودن کیفیت هوا استفاده می­شود.

### نمایش اطلاعات

جهت نمایش اطلاعات (دما و رطوبت گلخانه ، کیفیت هوا ، میزان آب موجود در مخزن و ...) از ماژول نمایشگر LCD16\*2 استفاده می­شود. در این پروژه جهت صرفه جویی در تعداد پین ها از ماژول LCD16\*2 به همراه رابط I2C استفاده می­شود. با استفاده از این ماژول فقط دو پین A5 و A4 به نمایشگر متصل می­شوند در حالی که اگر نمایشگر بدون ماژول I2C و به صورت مستقیم به برد Arduino UNO وصل شود ، حداقل 6 پین از برد باید به نمایشگر متصل شوند. همچنین از آنجایی که میزان اطلاعات قابل نمایش به نسبت زیاد است ، از یک دکمه جهت تغییر اطلاعات نمایش داده شده استفاده می­شود. بدین صورت که با فشردن دکمه ، اطلاعات قبلی نمایش داده شده (مانند دما) پاک شده و اطلاعات جدید ( مانند سرعت چرخش فن) نمایش داده می­شوند.

#include <Wire.h>

#include <OneWire.h>

#include <DallasTemperature.h>

از این سه کتابخانه برای اتصال به سنسور حرارتی DS18B20 استفاده می­شود.

#include <LiquidCrystal\_I2C.h>

از این کتابخانه جهت ایجاد اتصال I2C با نمایشگر استفاده می­شود.

#include <math.h>

این کتابخانه شامل تعدادی توابع و متغیر های مورد نیاز در محاسبات ریاضی است. در این پروژه مقدار عدد پی را با استفاده از این کتابخانه بدست میاید.

// Smart garden pins

ابتدا شماره پین های متصل به قطعات گلخانه هوشمند را در متغیر هایی ذخیره می­شوند.

int pump\_pin\_number = 13;

شماره پین متصل به پمپ در این متغیر ذخیره می­شود.

int soil\_water\_level\_pin = A3;

شماره پین متصل به سنسور رطوبت خاک در این متغیر ذخیره می­شوند.

int fan\_pin\_number = 11;

شماره پین متصل به فن در این متغیر ذخیره می­شوند.

int DS18B20\_pin\_number = 4;

شماره پین متصل به سنسور دما در این متغیر ذخیره می­شوند.

int UltraSonic\_Echo\_pin\_number = 10;

شماره پین متصل به پین Echo سنسور فراصوت در این متغیر ذخیره می­شوند.

int UltraSonic\_Trig\_pin\_number = 9;

شماره پین متصل به پین Trig سنسور فراصوت در این متغیر ذخیره می­شوند.

//home security pins

در خط های پایین شماره پین های متصل به قطعات امنیت خانه در متغیر ها ذخیره می­شوند.

int ReedSwitch\_pin\_number = 8;

شماره پین متصل به reed switch در این متغیر ذخیره می­شوند.

int MQ5\_pin\_number = A1;

شماره پین متصل به سنسور گازی در این متغیر ذخیره می­شوند.

int Buzzer\_pin\_number = 6;

شماره پین متصل به بازر در این متغیر ذخیره می­شوند.

int Lamp\_pin\_number = 7;

شماره پین متصل چراغ (رله متصل به لامپ) در این متغیر ذخیره می­شوند.

int PIR\_pin\_number = 5;

شماره پین متصل به سنسور PIR در این متغیر ذخیره می­شوند.

int Display\_Toggle\_Controll\_pin\_number = 4;

شماره پین متصل به دکمه ( جهت تغییر اطلاعات نمایش داده شده روی نمایشگر) در این متغیر ذخیره می­شوند.

//LCD Display pins

در خطوط زیر شماره پین های متصل به ماژول LCD در متغیر ها ذخیره می­شوند.

int SDA\_pin\_number = A4;

int Scl\_pin\_numberr = A5;

در خطوط پایین متغیر های کلی تعریف می­شوند.

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

از این متغیر جهت ارتباط با نمایشگر استفاده می­شود.

OneWire oneWire(DS18B20\_pin\_number);

DallasTemperature Temprature\_sensors(&oneWire);

از دو متغیر بالا جهت برقراری ارتباط با سنسور دما استفاده می­شود.

float Temperature\_C = 0.00;

دما به درجه سانتی گراد در این متغیر ذخیره می­شود.

int soil\_water\_level = 0;

میزان رطوبت خاک در این متغیر ذخیره می­شود.

bool is\_soil\_dry = false;

خشک بودن یا نبودن خاک در این متغیر ذخیره می­شود.

int fanSpeed = 0;

سرعت فن به صورت عددی بین 0 تا 255 در این متغیر ذخیره می­شود.

int fanLCD = 0;

سرعت فن به صورت درصدی در این متغیر ذخیره می­شود.

bool pump\_state = false;

روشن و یا خاموش بودن پمپ در این متغیر ذخیره می­شود.

float water\_level = 0;

میزان سطح آب به سانتی متر در این متغیر ذخیره می­شود.

float water\_volume = 0;

میزان آب موجود در مخزن به صورت سانتی متر مکعب ذخیره می­شود.

int air\_quality\_value = 0;

کیفیت هوا در این متغیر ذخیره می­شود.

bool is\_air\_breathable = false;

سالم و یا ناسالم بودن هوا در این متغیر ذخیره می­شود.

bool Reed\_Switch\_status = false;

وضعیت reed switch در این متغیر ذخیره می­شود.

bool is\_window\_open = false;

باز و یا بسته بودن پنجره در این متغیر ذخیره می­شود.

bool has\_detected\_movement = false;

تشخیص حرکت در این متغیر ذخیره می­شود.

bool was\_button\_active = false;

از این متغیر جهت تولید وضعیت روشن و خاموش (toggle) استفاده می­شود.

int Display\_Info\_Index = 0;

شماره اطلاعات نمایش داده شده در این متغیر ذخیره می­شود.

int Display\_Max\_Index = 6;

تعداد اطلاعات قابل نمایش در این متغیر ذخیره می­شود.

//controll variables

در آخر متغیر های کنترلی در تعریف می­شوند.

int thresholdValue = 800;

این متغیر سطح نامناسب بودن کیفیت هوا را مشخص می­کند.

int max\_temprature = 70;

در این متغیر سطح بالایی دما ذخیره می­شود.

int min\_temprature = 30;

در این متغیر سطح پایینی دما ذخیره می­شود.

int water\_tank\_radios = 5;

در این متغیر شعاع مخزن آب به سانتی متر ذخیره می­شود.

int water\_tank\_max\_height = 50;

در این متغیر ارتفاع مخزن آب (ارتفاع سنسور فراصوت) ذخیره می­شود.

void setup() {

در تابع setup نوع پین ها (ورودی ، خروجی) مشخص می­شود.

//setting up pinmodes

pinMode(pump\_pin\_number , OUTPUT);

pinMode(soil\_water\_level\_pin , INPUT);

pinMode(fan\_pin\_number , OUTPUT);

pinMode(DS18B20\_pin\_number , INPUT);

pinMode(ReedSwitch\_pin\_number , INPUT\_PULLUP);

pinMode(MQ5\_pin\_number , INPUT);

pinMode(Buzzer\_pin\_number , OUTPUT);

pinMode(UltraSonic\_Trig\_pin\_number, OUTPUT);

pinMode(UltraSonic\_Echo\_pin\_number, INPUT);

pinMode(has\_detected\_movement, INPUT);

pinMode(Lamp\_pin\_number, OUTPUT);

//starting the lcd

lcd.begin(); //Defining 16 columns and 2 rows of lcd display

ارتباط با ماژول نمایشگر برقرار می­شود.

lcd.backlight();//To Power ON the back light

چراغ پس زمینه نمایشگر روشن می­شود.

//starting serial com

Serial.begin(9600);

ارتباط سریال برقرار می­شود. از این ارتباط جهت ارسال اطلاعات به کامپیوتر استفاده می­شود.

}

از آنجایی که این پروژه به نسبت پروژه های قبلی پپیچیده تر و دارای قسمت های مختلفی است ، دستورات را دسته بندی شده و در توابع مختلف قرار می­گیرد. در تابع loop توابع تعریف شده به ترتیب اجرا می­شوند. ابتدا توابع مرتبط با دریافت اطلاعات از سنسور ها سپس توابع مرتبط با بررسی و واکنش به اطلاعات دریافت شده از سنسورها و در آخر توابع مربوط به نمایش اطلاعات با استفاده از ماژول نمایشگر اجرا می­شوند. و در آخر تاخیر نیم ثانیه ای قرار داده می­شود.

void loop() {

//gathering all the data

Get\_temprature();

Get\_soil\_status();

Get\_water\_level();

Get\_air\_quality();

Get\_Reed\_Switch();

Get\_Movement();

//controlling the envirment

Fan\_Controll();

Pump\_controll();

Control\_Air\_quality();

Security\_Controll();

Lamp\_Controll();

//Display on LCD

Toggle\_Controll();

Displayall();

delay(500);

}

حال به بررسی تمامی تابع ها پرداخته می­شود.

void Get\_temprature() {

این تابع وظیفه بدست آوردن دما را برعهده دارد.

Serial.print(" Requesting temperatures...");

Temprature\_sensors.requestTemperatures();

درخواست اندازه­گیری دما به سنسور ارسال می­شود.

Serial.println("DONE");

Serial.print("Temperature is: ");

Temperature\_C = Temprature\_sensors.getTempCByIndex(0);

دما اندازه گیری شده به درجه سانتی گراد در متغیر Temperature\_C ذخیره می­شود.

Serial.print(Temperature\_C);

دما به کامپیوتر ارسال می­شود.

}

void Get\_soil\_status() {

این تابه وظیفه اندازه گیری رطوبت خاک را بر عهده دارد.

int soil\_water\_level = analogRead(soil\_water\_level\_pin);

ابتدا ولتاژ خروجی سنسور اندازه­گیری می­شود.

Serial.print(soil\_water\_level);

if (soil\_water\_level < thresholdValue) {

اگر رطوبت از سطح آستانه خشکی کمتر بود:

Serial.println(" - Doesn't need watering");

is\_soil\_dry = false;

وضعیت خشکی خاک در متغیر ذخیره می­شود.

}

else {

اگر رطوبت از سطح آستانه خشکی بیشتر بود:

Serial.println(" - Time to water your plant");

is\_soil\_dry = true;

وضعیت تر بودن خاک در متغیر ذخیره می­شود.

}

}

void Fan\_Controll() {

این تابع وظیفه کنترل سرعت فن را بر عهده دارد.

if (Temperature\_C < min\_temprature) { // if temp is lower than minimum temp

اگر دما از سطح آستانه پایین کمتر بود:

fanSpeed = 0; // fan is not spinning

عدد 0 به عنوان سرعت فن در متغیر ذخیره می­شود.

digitalWrite(fan\_pin\_number, LOW);

فن متوقف می­شود.

}

if (Temperature\_C > max\_temprature) { // if temp is higher than maximum temp

اگر دما از حد آستانه بالا بیشتر بود:

fanSpeed = 255; // fan is spinning

عدد 255 به عنوان سرعت فن در متغیر ذخیره می­شود.

digitalWrite(fan\_pin\_number, HIGH);

فن با بیشترین سرعت می­چرخد.

}

if ((Temperature\_C >= min\_temprature) && (Temperature\_C <= max\_temprature)) {

اگر دما بین آستانه پایین و بالا بود:

fanSpeed = map(Temperature\_C, min\_temprature, max\_temprature, 32, 255);

سرعت فن مطابق با دما محاسبه می­شود و در متغیر fanSpeed ذخیره می­شود.

fanLCD = map(Temperature\_C, min\_temprature, max\_temprature, 0, 100);

سرعت فن به درصد محاسبه و در fanLCD ذخیره می­شود.

analogWrite(fan\_pin\_number, fanSpeed); // spin the fan at the fanSpeed speed

سرعت فن برابر با مقدار محاسبه شده و ذخیره شده در fanSpeed قرار می­گیرد.

}

}

void Get\_water\_level() {

این تابع میزان آب موجود در مخزن را اندازه­گیری می­کند.

long t = 0, h = 0, hp = 0;

digitalWrite(UltraSonic\_Trig\_pin\_number, LOW);

ابتدا با LOW شدن پین Trig ، سنسور برای ارسال موج آماده می­شود.

delayMicroseconds(2);

digitalWrite(UltraSonic\_Trig\_pin\_number, HIGH);

با HIGH شدن پین Trig ، سنسور موج فراصوت را ارسال می­کند.

delayMicroseconds(10);

digitalWrite(UltraSonic\_Trig\_pin\_number, LOW);

با LOW شدن دوباره پین Trig، ارسال موج فراصوت متوقف می­شود.

long duration = pulseIn(UltraSonic\_Echo\_pin\_number, HIGH);

با استفاده از دستور pulseIn مدت زمان رفت و برگشت موج اندازه­گیری می­شود.

int distance = duration \* 0.034 / 2;

با استفاده از سرعت موج و زمان رفت و برگشت ، فاصله سنسور تا جسم مقابلش (سطح آب) اندازه گیری می­شود.

water\_level = water\_tank\_max\_height - distance;

ارتفاع آب با استفاده از فاصله سنسور تا سطح آب و ارتفاع مخزن (سنسور) اندازه­گیری می­شود.

water\_volume = M\_PI \* pow(water\_tank\_radios, 2) \* water\_level;

حجم آب به سانتی­متر مکعب اندازه گیری و در متغیر ذخیره می­شود.

}

void Pump\_controll() {

این تابع وظیفه کنترل پمپ آب را بر عهده دارد.

if (is\_soil\_dry) {

اگر خاک خشک بود:

digitalWrite(pump\_pin\_number , HIGH);

پمپ روشن می­شود.

pump\_state = true;

وضعیت روشن بودن پمپ در متغیر ذخیره می­شود.

} else {

اگر خاک تر بود:

digitalWrite(pump\_pin\_number , LOW);

pump\_state = false;

پمپ خاموش می­شود و وضعیت خاموش بودن پمپ در متغیر ذخیره می­شود.

}

}

void Get\_air\_quality() {

در این تابع کیفیت هوا محاسبه می­شود.

air\_quality\_value = analogRead(MQ5\_pin\_number);

ابتدا ولتاژ خروجی سنسور اندازه­گیری می­شود.

Serial.println("air quality : " + String(air\_quality\_value));

if (air\_quality\_value <= 800) {

اگر ولتاژ خروجی سنسور از آستانه کمتر بود:

is\_air\_breathable = true;

مناسب بودن کیفیت هوا در متغیر ذخیره می­شود.

Serial.println("Air is breathable");

}

else {

در غیر اینصورت:

is\_air\_breathable = false;

نامناسب بودن کیفیت هوا در متغیر ذخیره می­شود.

Serial.println("Air is not breathable - gas leak dettected");

}

}

void Control\_Air\_quality() {

این تابع وظیفه اعلام کیفیت نامناسب هوا را برعهده دارد.

if (!is\_air\_breathable) {

اگر کیفیت هوا مناسب نبود:

tone(Buzzer\_pin\_number, 1000);

از بازر صدایی با فرکانس 1000 هرتز پخش می­شود.

delay(2000);

تاخیر 2 ثانیه ای قرار می­گیرد.

noTone(Buzzer\_pin\_number);

صدای بازر قطع می­شود.

}

}

void Get\_Reed\_Switch() {

با استفاده از این تابع وضعیت سنسور reed switch بدست می­آید.

Reed\_Switch\_status = digitalRead(ReedSwitch\_pin\_number);

ابتدا وضعیت سنسور (وصل بودن) بدست می­آید.

if (Reed\_Switch\_status == HIGH) {

اگر سنسور وصل باشد:

is\_window\_open = true;

وضعیت باز بودن پنجره در متغیر ذخیره می­شود.

} else {

در غیر اینصورت:

is\_window\_open = false;

وضعیت بسته بودن پنجره در متغیر ذخیره می­شود.

}

}

void Security\_Controll() {

این تابع وظیفه امنیت خانه را برعهده دارد.

if (!is\_window\_open) {

اگر پنجره باز باشد:

tone(Buzzer\_pin\_number, 2500);

بازر به صدا در می­آید.

delay(1000);

noTone(Buzzer\_pin\_number);

صدای بازر پس از 1 ثانیه قطع می­شود.

}

}

void Get\_Movement() {

این تابع وظیفه گرفتن اطلاعات از سنسور PIR را بر عهده دارد.

int movement\_status = digitalRead(PIR\_pin\_number);

وضعیت بودن و یا نبودن حرکت از سنسور گرفته می­شود.

if (movement\_status == HIGH) {

اگر حرکت تشخیص داده شده باشد:

has\_detected\_movement = true;

وجود حرکت در متغیر ذخیره می­شود.

} else {

در غیر اینصورت:

has\_detected\_movement = false;

نبود حرکت در متغیر ذخیره می­شود.

}

}

void Lamp\_Controll() {

این تابع روشن و خاموش شدن چراغ را کنترل می­کند.

if (has\_detected\_movement) {

اگر حرکتی شناسایی شده باشد:

digitalWrite(Lamp\_pin\_number, HIGH);

چراغ روشن می­شود.

} else {

در غیر اینصورت:

digitalWrite(Lamp\_pin\_number, LOW);

چراغ خاموش می­شود.

}

}

### نمایش اطلاعات بر روی نمایشگر

ارتفاع آب درون مخزن ، حجم آب درون مخزن ، دمای گلخانه ، سرعت فن ، کیفیت هوا ، وضعیت پنجره و وضعیت پمپ آب مجموعه اطلاعاتی هستند که بر روی نمایشگر نشان داده می­شوند. از آنجایی که نمایشگر فقط میتواند 32 کارکتر را بر روی صفحه خود نمایش دهد ، تمامی اطلاعات نمیتوانند به صورت همزمان بر روی صفحه نمایشگر ظاهر شوند. برای حل این مسئله فقط یک مورد از مجموعه اطلاعات را در هر زمان بر روی نمایشگر نمایش داده می­شود و دکمه ای بر روی مدار قرار داده می­شود که با فشردن آن اطلاعات نمایش داده شده تغییر می­کنند.

void Toggle\_Controll() {

این تابع وظیفه تغییر اطلاعات نمایش داده شده بر روی نمایشگر را بر عهده دارد.

int button\_state = digitalRead(Display\_Toggle\_Controll\_pin\_number);

ابتدا وضعیت فشرده شدن دکمه در متغیر ذخیره می­شود.

if (was\_button\_active == false) {

اگر دکمه قبلا فشرده نشده باشد:

if (button\_state == LOW) {

اگر دکمه در حال حاضر فشرده شده باشد:

was\_button\_active = true;

فشرده شدن دکمه در متغیر ذخیره می­شود.

if(Display\_Info\_Index >= Display\_Max\_Index){

اگر آخرین مورد از مجموعه اطلاعات در حال نمایش است:

Display\_Info\_Index = 0;

اولین مورد از مجموعه اطلاعات نمایش داده می­شود.

} else{

در غیر اینصورت:

Display\_Info\_Index += 1;

مورد بعدی از مجموعه اطلاعات نمایش داده می­شود.

}

}

}

if (was\_button\_active == true) {

اگر دکمه قبلا فشرده شده باشد:

if (button\_state == HIGH) {

اگر دکمه در حال حاضر فشرده نشده باشد:

was\_button\_active = false;

فشرده نشدن دکمه در متغیر ذخیره می­شود.

}

}

}

void Displayall() {

این تابع وظیفه ی نمایش اطلاعات بر روی نمایشگر را بر عهده دارد.

switch(Display\_Info\_Index){

با استفاده از دستور switch اطلاعاتی را که باید نمایش داده شوند (طبق عدد ذخیره شده در متغیر Display\_Info\_Index که توسط تابع قبلی کنترل می­شود) انتخاب می­شوند.

case 0:

اگر ارتفاع آب انتخاب شد:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به اولین محل انتقال پیدا می­کند.

lcd.print("Water Level:");

نوشته Water Level نمایش داده می­شود.

lcd.setCursor(0,1);

مکان نما به ستون دوم انتقال پیدا می­کند.

lcd.print(String(water\_level) + " cm");

ارتفاع آب به سانتی متر نوشته می­شود.

break;

case 1:

اگر حجم آب انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("Water Volume:");

نوشته Water Volume نوشته می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(water\_volume) + " cm^3");

حجم آب به سانتی متر مکعب نوشته می­شود.

break;

case 2:

اگر دما انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("Temprature:");

نوشته Temprature نوشته می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(Temperature\_C) + " C");

دما به سانتی گراد نمایش داده می­شود.

break;

case 3:

اگر سرعت فن انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("Fan Speed:");

نوشته Fan Speed: نوشته می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(fanLCD) + " %");

سرعت فن به درصد نوشته می­شود.

break;

case 4:

اگر کیفیت هوا انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("Air quality:");

نوشتهAir quality: نوشته می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(air\_quality\_value));

کیفیت هوا نمایش داده می­شود.

break;

case 5:

اگر وضعیت پنجره انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("Window status:");

نوشته ی Window status: نمایش داده می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(is\_window\_open));

وضعیت پنجره نمایشداده می­شود.

break;

case 6:

و اگر وضعیت پمپ انتخاب شود:

lcd.clear();

صفحه نمایش پاک می­شود.

lcd.setCursor(0,0);

مکان نما به خانه اول انتقال پیدا می­کند.

lcd.print("pump status:");

نوشته ی pump status: نمایش داده می­شود.

lcd.setCursor(0,1);

مکان نما به سطر دوم انتقال پیدا می­کند.

lcd.print(String(pump\_state) + "");

وضعیت پمپ نمایش داده می­شود.

break;

}