<https://iotprojectsideas.com/fingerprint-door-lock-security-systems-using-arduino-lcd/>

Biometric Security System with Fingerprint Sensor & Arduino

<https://diyprojectslab.com/biometric-security-system-with-fingerprint-sensor/>

Trong dự án này, chúng tôi sẽ xây dựng Hệ thống bảo mật sinh trắc học với Arduino & cảm biến vân tay. Hệ thống này sẽ có thể nhận dạng ai đó bằng dấu vân tay của họ và cho phép họ vào khu vực an toàn. Chúng tôi sẽ sử dụng mô-đun cảm biến vân tay R307 để thu thập và lưu trữ dữ liệu vân tay.

Dữ liệu này sẽ được sử dụng để xác thực danh tính của một người khi người đó cố gắng truy cập vào khu vực được bảo mật. Hệ thống sẽ có thể từ chối hoặc truy cập người đó dựa trên **dữ liệu dấu vân tay** của họ .

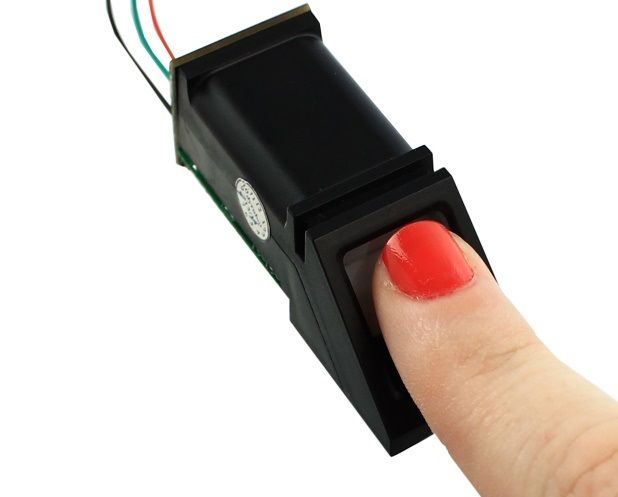
**Cảm ơn bạn, NextPCB!**

Dự án của chúng tôi đã hoàn thành thành công nhờ sự trợ giúp hữu ích của [**NextPCB**](https://www.nextpcb.com/) . Nếu bạn đang làm việc trong một **dự án PCB** , chúng tôi khuyên bạn nên truy cập trang web của họ để tận dụng một số phiếu giảm giá và giảm giá tuyệt vời.

Hệ thống sẽ được trang bị màn hình LCD để hiển thị các thông báo trạng thái và đèn LED để cung cấp phản hồi. Kiểm tra bài viết trước của tôi về [Hệ thống an ninh gia đình dựa trên GSM sử dụng cảm biến siêu âm](https://diyprojectslab.com/gsm-security-system-ultrasonic-sensor/)

### Vật liệu cần thiết

* bo mạch Arduino
* Mô-đun cảm biến vân tay
* Màn hình LCD I2C 16×2
* Mô-đun rơle
* Nút ấn
* Đèn LED (tùy chọn, để phản hồi trực quan)
* Dây nhảy
* Bánh mì;

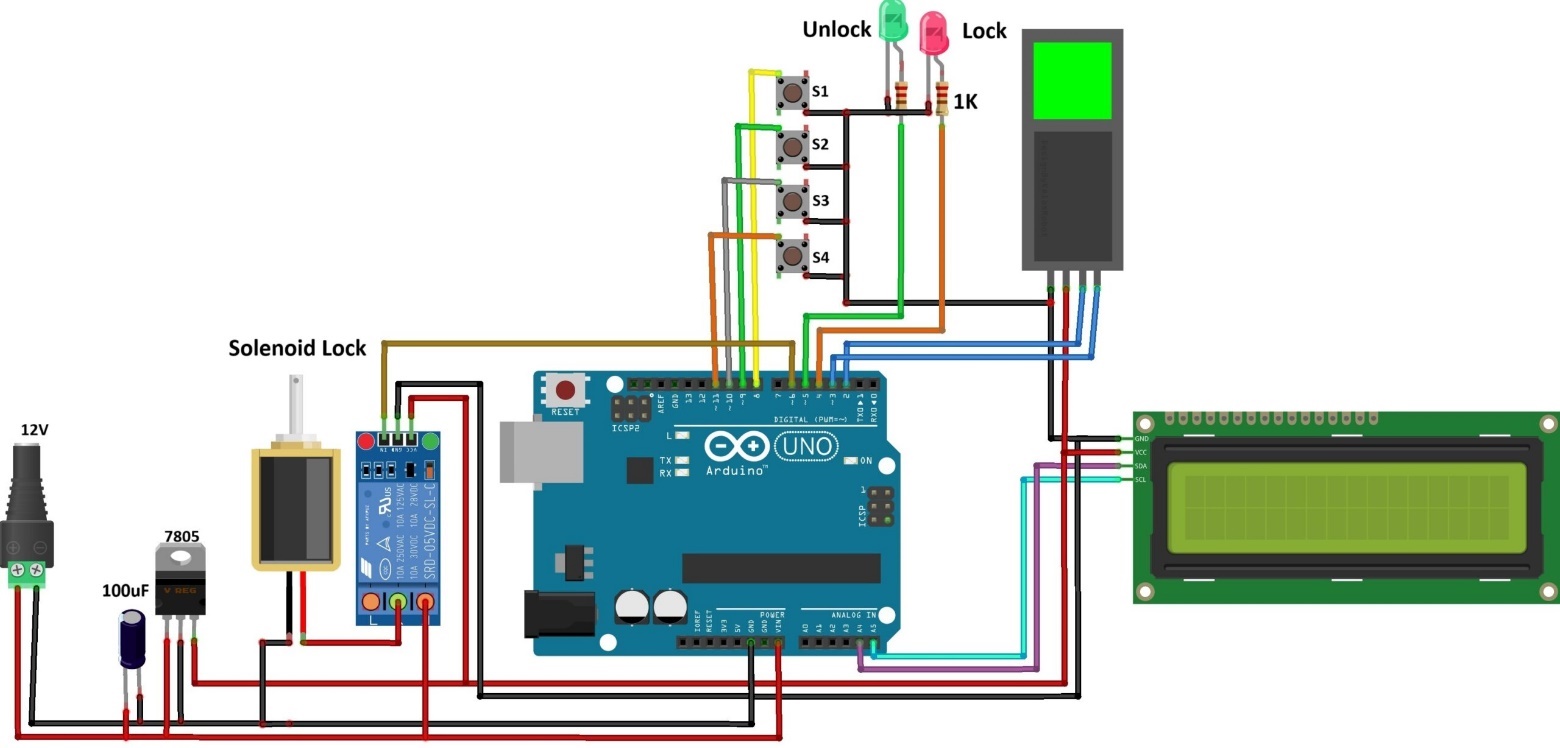
[](https://diyprojectslab.com/wp-content/uploads/2023/04/R307-Optical-Fingerprint-Sensor.jpg)

Hệ [**thống bảo mật sinh trắc học**](https://diyprojectslab.com/fingerprint-door-lock-security-arduino/) đảm bảo truy cập an toàn bằng dấu vân tay. Nó sử dụng [mô-đun vân tay R307](https://diyprojectslab.com/r307-fingerprint-sensor-with-arduino/) , Arduino, rơle, khóa điện từ, LCD và các nút. Người dùng tương tác thông qua các nút ấn để **đăng ký** , **xóa** và **quét** . Nhấn “ **Quét** ” sẽ nhắc đặt ngón tay lên mô-đun. Nó quét, gửi dữ liệu tới Arduino và xác minh dựa trên dấu vân tay được lưu trữ.

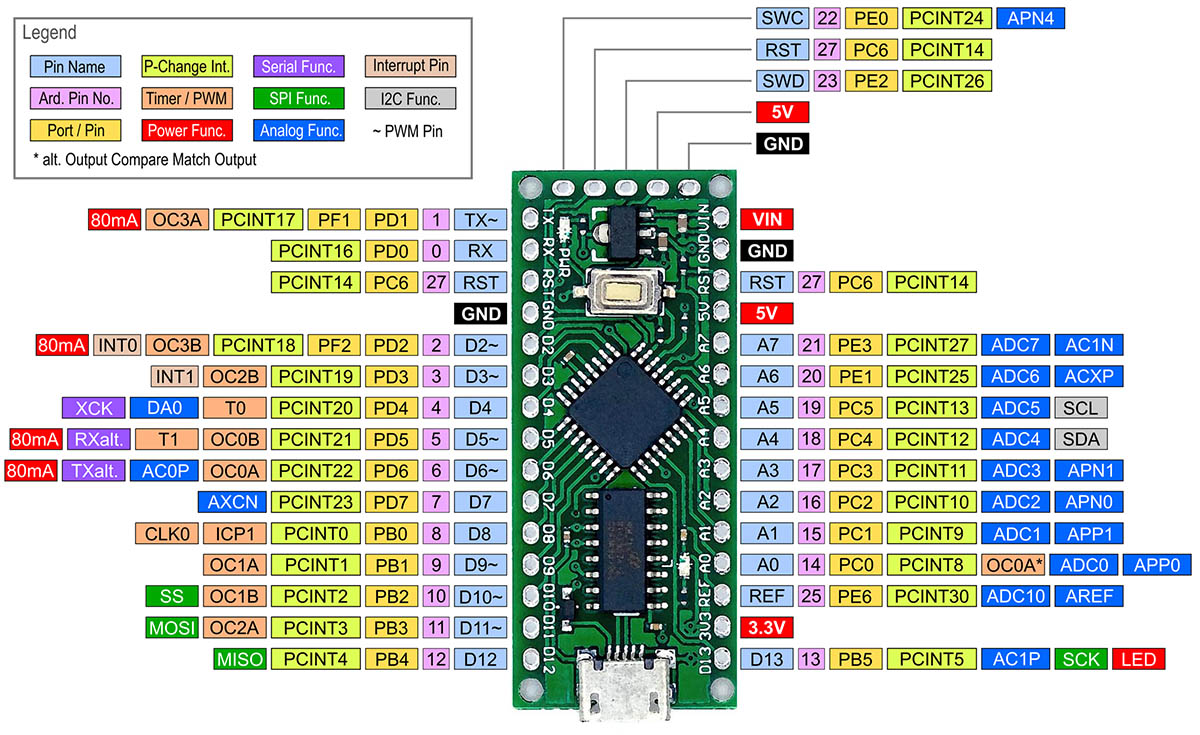
Một trận đấu sẽ kích hoạt rơle, do đó cấp nguồn cho khóa điện từ để mở khóa. Màn hình LCD cung cấp phản hồi của người dùng, chẳng hạn như “ **Quyền truy cập được cấp** ” hoặc “ **Quyền truy cập bị từ chối** ”.

### **Sơ đồ mạch & Kết nối – Hệ thống bảo mật sinh trắc học**

Đây là sơ đồ mạch của **Hệ thống bảo mật sinh trắc học với cảm biến vân tay** & Arduino được thiết kế bằng Phần mềm Fritzing:

[](https://diyprojectslab.com/wp-content/uploads/Circuit-Biometric-Security-System-with-Fingerprint-Sensor-Arduino-scaled.jpg)

Dự án bao gồm nhiều thành phần khác nhau phối hợp với nhau để tạo ra kết quả. Chúng ta hãy xem xét sâu hơn về mối liên hệ giữa các thành phần khác nhau trong dự án này.



**FPM10 Red 3.3V – Black GND – Green TX – Yellow RX**

**Kết nối cảm biến vân tay:**

* **TX** sang Arduino **D2**
* **RX** sang Arduino **D3**
* **VCC** sang Arduino **5V**
* **GND** sang Arduino **GND**

**Kết nối LCD bằng I2C:**

* **SDA** sang Arduino **A4**
* **SCL** sang Arduino **A5**
* **VCC** sang Arduino **5V**
* **GND** sang Arduino **GND**

**Kết nối các nút bấm: (đăng ký, xóa, lên, xuống)**

* Kết nối một bên với các chân kỹ thuật số ( **8** , **9** , **10, 11)**
* Kết nối phía bên kia với **GND**

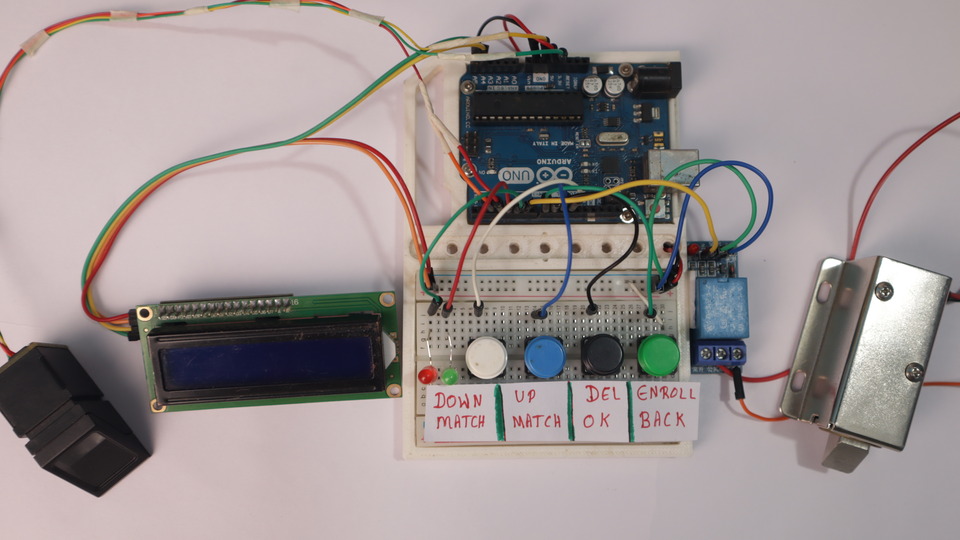
**Kết nối đèn LED cho đèn báo cửa: (Cửa mở, Cửa đóng):**

* Đèn LED mở cửa:
  + Chân dài hơn tới chân kỹ thuật số **4**
  + Chân ngắn hơn tới **GND**
* Đèn LED đóng cửa:
  + Chân dài hơn tới chân kỹ thuật số **5**
  + Chân ngắn hơn tới **GND**

**Kết nối mô-đun chuyển tiếp:**

* **Chân IN** sang chân số **6**
* **VCC** sang Arduino **5V**
* **GND** sang Arduino **GND**
* **COM** : Đã kết nối với Khóa điện từ
* **NC** : Đã kết nối với **12V**

Những hướng dẫn đơn giản này sẽ giúp bạn hiểu rõ cách kết nối các thành phần phần cứng với Arduino cho hệ thống bảo mật sinh trắc học của bạn.

[](https://diyprojectslab.com/wp-content/uploads/Circuit-On-breadboard-1.jpg)

Hệ thống chạy bằng nguồn điện DC 12V. Bộ điều chỉnh điện áp 7805 cung cấp **5V** cho Cảm biến vân tay, rơle và Arduino UNO. Tụ **điện điện phân 100uF/35V** cung cấp điện áp ổn định ở các chân nguồn điện.

### **Cài đặt thư viện cần thiết**

Hệ thống bảo mật sinh trắc học với mã cảm biến vân tay, được viết bằng Arduino IDE, phụ thuộc vào hai thư viện để biên dịch. Trước tiên hãy tải xuống thư viện đòn và thêm nó vào Arduino IDE của bạn:

1. [**Thư viện cảm biến vân tay Adafbean**](https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library)
2. [**Thư viện LCD LiquidCrystal I2C**](https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library)

Đi tới “ **Phác thảo** “ > “ **Bao gồm Thư viện** ” > “ **Quản lý Thư viện** ,” sau đó tìm kiếm và cài đặt thư viện “\*\*\*\*\*\*\*\*”.

### **Tải mã lên**

Sao chép mã bên dưới và dán mã được cung cấp vào Arduino IDE.

#include <Adafruit\_Fingerprint.h>

#include <SoftwareSerial.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 16, 2); // Initialize the LCD

SoftwareSerial fingerPrint(2, 3);

// Define constants for pins

#define enroll 8

#define del 9

#define up 10

#define down 11

#define Door\_Open 4

#define Door\_Close 5

#define RELAY\_PIN 6

uint8\_t id;

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&fingerPrint);

void setup()

{

// Initialize pins

pinMode(RELAY\_PIN, OUTPUT);

digitalWrite(RELAY\_PIN, LOW); // Initially, relay is off

pinMode(enroll, INPUT\_PULLUP);

pinMode(up, INPUT\_PULLUP);

pinMode(down, INPUT\_PULLUP);

pinMode(del, INPUT\_PULLUP);

pinMode(Door\_Open, OUTPUT);

pinMode(Door\_Close, OUTPUT);

// Initialize LCD

lcd.init();

lcd.backlight();

lcd.print("Fingerprint");

lcd.setCursor(0, 1);

lcd.print("Security System");

delay(2000);

lcd.clear();

Serial.begin(9600);

finger.begin(57600);

lcd.clear();

lcd.print("Finding Module");

lcd.setCursor(0, 1);

delay(1000);

if (finger.verifyPassword())

{

Serial.println("Found fingerprint sensor!");

lcd.clear();

lcd.print("Found Module ");

delay(1000);

}

else

{

Serial.println("Did not find fingerprint sensor :(");

lcd.clear();

lcd.print("module not Found");

lcd.setCursor(0, 1);

lcd.print("Check Connections");

while (1);

}

}

void loop()

{

lcd.setCursor(0, 0);

lcd.print("Press UP/Down ");

lcd.setCursor(0, 1);

lcd.print("to start System");

digitalWrite(Door\_Close, HIGH);

if (digitalRead(up) == 0 || digitalRead(down) == 0)

{

for (int i = 0; i < 5; i++)

{

lcd.clear();

lcd.print("Place Finger");

delay(2000);

int result = getFingerprintIDez();

if (result >= 0)

{

digitalWrite(Door\_Open, HIGH);

digitalWrite(Door\_Close, LOW);

lcd.clear();

lcd.print("Allowed");

lcd.setCursor(0, 1);

lcd.print("Gate Opened ");

digitalWrite(RELAY\_PIN, HIGH);

delay(5000);

digitalWrite(RELAY\_PIN, LOW);

digitalWrite(Door\_Close, HIGH);

digitalWrite(Door\_Open, LOW);

lcd.setCursor(0, 1);

lcd.print("Gate Closed ");

return;

}

}

}

checkKeys();

delay(1000);

}

void checkKeys()

{

if (digitalRead(enroll) == 0)

{

lcd.clear();

lcd.print("Please Wait");

delay(1000);

while (digitalRead(enroll) == 0);

Enroll();

}

else if (digitalRead(del) == 0)

{

lcd.clear();

lcd.print("Please Wait");

delay(1000);

delet();

}

}

void Enroll()

{

int count = 0;

lcd.clear();

lcd.print("Enroll Finger ");

lcd.setCursor(0, 1);

lcd.print("Location:");

while (1)

{

lcd.setCursor(9, 1);

lcd.print(count);

if (digitalRead(up) == 0)

{

count++;

if (count > 25)

count = 0;

delay(500);

}

else if (digitalRead(down) == 0)

{

count--;

if (count < 0)

count = 25;

delay(500);

}

else if (digitalRead(del) == 0)

{

id = count;

getFingerprintEnroll();

return;

}

else if (digitalRead(enroll) == 0)

{

return;

}

}

}

void delet()

{

int count = 0;

lcd.clear();

lcd.print("Delete Finger ");

lcd.setCursor(0, 1);

lcd.print("Location:");

while (1)

{

lcd.setCursor(9, 1);

lcd.print(count);

if (digitalRead(up) == 0)

{

count++;

if (count > 25)

count = 0;

delay(500);

}

else if (digitalRead(down) == 0)

{

count--;

if (count < 0)

count = 25;

delay(500);

}

else if (digitalRead(del) == 0)

{

id = count;

deleteFingerprint(id);

return;

}

else if (digitalRead(enroll) == 0)

{

return;

}

}

}

uint8\_t getFingerprintEnroll()

{

int p = -1;

lcd.clear();

lcd.print("finger ID:");

lcd.print(id);

lcd.setCursor(0, 1);

lcd.print("Place Finger");

delay(2000);

while (p != FINGERPRINT\_OK)

{

p = finger.getImage();

switch (p)

{

case FINGERPRINT\_OK:

Serial.println("Image taken");

lcd.clear();

lcd.print("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No Finger");

lcd.clear();

lcd.print("No Finger");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

lcd.clear();

lcd.print("Comm Error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

lcd.clear();

lcd.print("Imaging Error");

break;

default:

Serial.println("Unknown error");

lcd.clear();

lcd.print("Unknown Error");

break;

}

}

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

lcd.clear();

lcd.print("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

lcd.clear();

lcd.print("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

lcd.clear();

lcd.print("Comm Error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

lcd.clear();

lcd.print("Feature Not Found");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

lcd.clear();

lcd.print("Feature Not Found");

return p;

default:

Serial.println("Unknown error");

lcd.clear();

lcd.print("Unknown Error");

return p;

}

Serial.println("Remove finger");

lcd.clear();

lcd.print("Remove Finger");

delay(2000);

p = 0;

while (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

Serial.print("ID "); Serial.println(id);

p = -1;

Serial.println("Place same finger again");

lcd.clear();

lcd.print("Place Finger");

lcd.setCursor(0, 1);

lcd.print(" Again");

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.print(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

return;

}

}

p = finger.image2Tz(2);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();

if (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_ENROLLMISMATCH) {

Serial.println("Fingerprints did not match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

Serial.print("ID "); Serial.println(id);

p = finger.storeModel(id);

if (p == FINGERPRINT\_OK) {

Serial.println("Stored!");

lcd.clear();

lcd.print("Stored!");

delay(2000);

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_BADLOCATION) {

Serial.println("Could not store in that location");

return p;

} else if (p == FINGERPRINT\_FLASHERR) {

Serial.println("Error writing to flash");

return p;

} else {

Serial.println("Unknown error");

return p;

}

}

int getFingerprintIDez()

{

uint8\_t p = finger.getImage();

if (p != FINGERPRINT\_OK)

return -1;

p = finger.image2Tz();

if (p != FINGERPRINT\_OK)

return -1;

p = finger.fingerFastSearch();

if (p != FINGERPRINT\_OK)

{

lcd.clear();

lcd.print("Finger Not Found");

lcd.setCursor(0, 1);

lcd.print("Try Later");

delay(2000);

return -1;

}

// found a match!

Serial.print("Found ID #");

Serial.print(finger.fingerID);

return finger.fingerID;

}

uint8\_t deleteFingerprint(uint8\_t id)

{

uint8\_t p = -1;

lcd.clear();

lcd.print("Please wait");

p = finger.deleteModel(id);

if (p == FINGERPRINT\_OK)

{

Serial.println("Deleted!");

lcd.clear();

lcd.print("Figer Deleted");

lcd.setCursor(0, 1);

lcd.print("Successfully");

delay(1000);

}

else

{

Serial.print("Something Wrong");

lcd.clear();

lcd.print("Something Wrong");

lcd.setCursor(0, 1);

lcd.print("Try Again Later");

delay(2000);

return p;

}

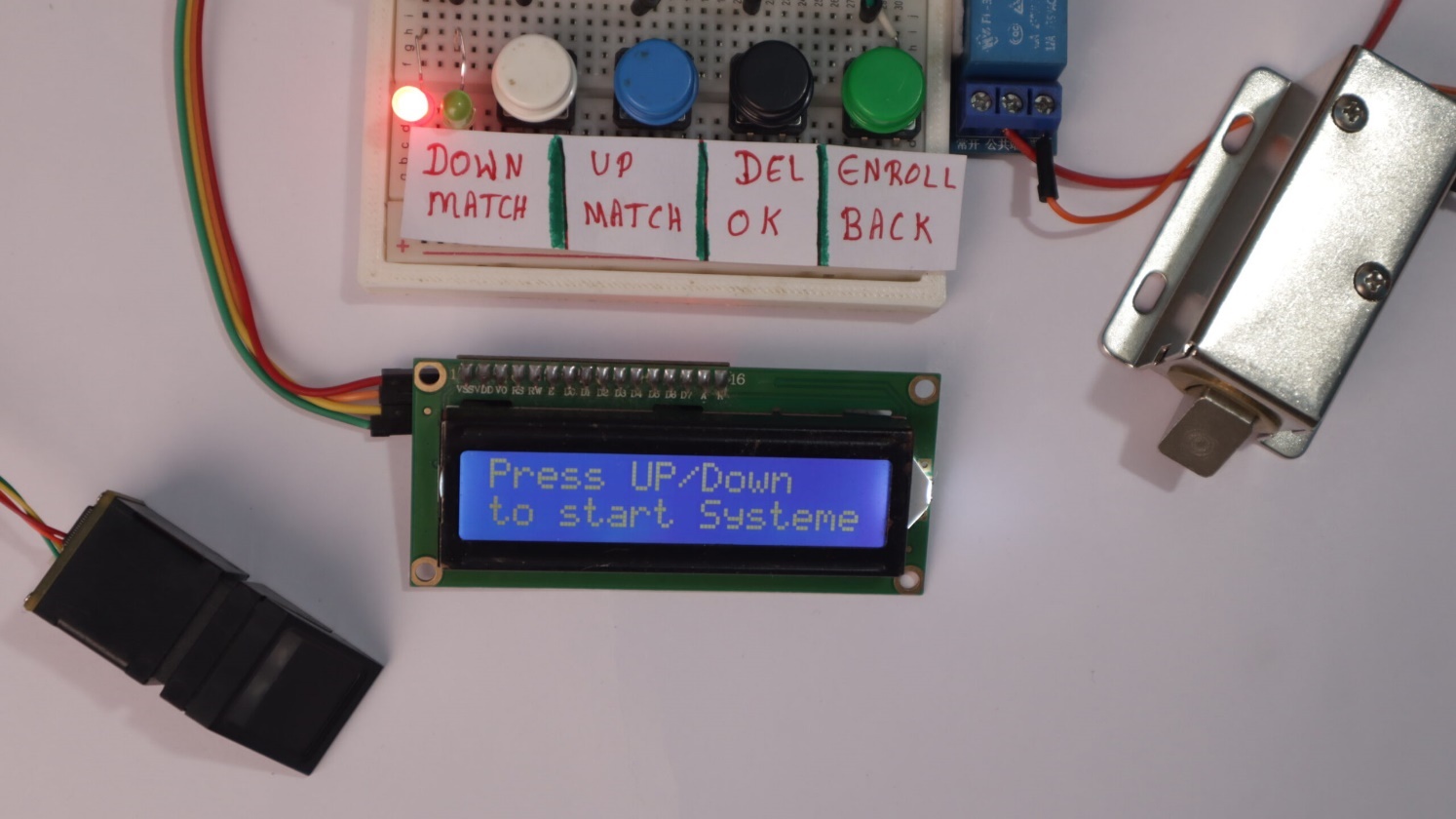
}

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289  290  291  292  293  294  295  296  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312  313  314  315  316  317  318  319  320  321  322  323  324  325  326  327  328  329  330  331  332  333  334  335  336  337  338  339  340  341  342  343  344  345  346  347  348  349  350  351  352  353  354  355  356  357  358  359  360  361  362  363  364  365  366  367  368  369  370  371  372  373  374  375  376  377  378  379  380  381  382  383  384  385  386  387  388  389  390  391  392  393  394  395  396  397  398  399  400  401  402  403  404  405  406  407  408  409  410  411  412  413  414  415  416  417  418  419  420  421  422  423  424  425  426  427  428  429  430  431  432  433  434 | #include <Adafruit\_Fingerprint.h>  #include <SoftwareSerial.h>  #include <LiquidCrystal\_I2C.h>    LiquidCrystal\_I2C lcd(0x27, 16, 2); // Initialize the LCD  SoftwareSerial fingerPrint(2, 3);    // Define constants for pins  #define enroll 8  #define del 9  #define up 10  #define down 11  #define Door\_Open 4  #define Door\_Close 5  #define RELAY\_PIN 6    uint8\_t id;  Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&fingerPrint);    void setup()  {    // Initialize pins    pinMode(RELAY\_PIN, OUTPUT);    digitalWrite(RELAY\_PIN, LOW); // Initially, relay is off    pinMode(enroll, INPUT\_PULLUP);    pinMode(up, INPUT\_PULLUP);    pinMode(down, INPUT\_PULLUP);    pinMode(del, INPUT\_PULLUP);    pinMode(Door\_Open, OUTPUT);    pinMode(Door\_Close, OUTPUT);      // Initialize LCD    lcd.init();    lcd.backlight();    lcd.print("Fingerprint");    lcd.setCursor(0, 1);    lcd.print("Security System");    delay(2000);    lcd.clear();      Serial.begin(9600);    finger.begin(57600);      lcd.clear();    lcd.print("Finding Module");    lcd.setCursor(0, 1);    delay(1000);      if (finger.verifyPassword())    {      Serial.println("Found fingerprint sensor!");      lcd.clear();      lcd.print("Found Module ");      delay(1000);    }    else    {      Serial.println("Did not find fingerprint sensor :(");      lcd.clear();      lcd.print("module not Found");      lcd.setCursor(0, 1);      lcd.print("Check Connections");      while (1);    }  }    void loop()  {    lcd.setCursor(0, 0);    lcd.print("Press UP/Down ");    lcd.setCursor(0, 1);    lcd.print("to start System");      digitalWrite(Door\_Close, HIGH);    if (digitalRead(up) == 0 || digitalRead(down) == 0)    {      for (int i = 0; i < 5; i++)      {        lcd.clear();        lcd.print("Place Finger");        delay(2000);        int result = getFingerprintIDez();        if (result >= 0)        {          digitalWrite(Door\_Open, HIGH);          digitalWrite(Door\_Close, LOW);            lcd.clear();          lcd.print("Allowed");          lcd.setCursor(0, 1);          lcd.print("Gate Opened   ");            digitalWrite(RELAY\_PIN, HIGH);          delay(5000);          digitalWrite(RELAY\_PIN, LOW);          digitalWrite(Door\_Close, HIGH);          digitalWrite(Door\_Open, LOW);            lcd.setCursor(0, 1);          lcd.print("Gate Closed   ");            return;        }      }    }    checkKeys();    delay(1000);  }    void checkKeys()  {    if (digitalRead(enroll) == 0)    {      lcd.clear();      lcd.print("Please Wait");      delay(1000);      while (digitalRead(enroll) == 0);      Enroll();    }    else if (digitalRead(del) == 0)    {      lcd.clear();      lcd.print("Please Wait");      delay(1000);      delet();    }  }    void Enroll()  {    int count = 0;    lcd.clear();    lcd.print("Enroll Finger    ");    lcd.setCursor(0, 1);    lcd.print("Location:");    while (1)    {      lcd.setCursor(9, 1);      lcd.print(count);      if (digitalRead(up) == 0)      {        count++;        if (count > 25)          count = 0;        delay(500);      }      else if (digitalRead(down) == 0)      {        count--;        if (count < 0)          count = 25;        delay(500);      }      else if (digitalRead(del) == 0)      {        id = count;        getFingerprintEnroll();        return;      }      else if (digitalRead(enroll) == 0)      {        return;      }    }  }    void delet()  {    int count = 0;    lcd.clear();    lcd.print("Delete Finger    ");    lcd.setCursor(0, 1);    lcd.print("Location:");      while (1)    {      lcd.setCursor(9, 1);      lcd.print(count);      if (digitalRead(up) == 0)      {        count++;        if (count > 25)          count = 0;        delay(500);      }      else if (digitalRead(down) == 0)      {        count--;        if (count < 0)          count = 25;        delay(500);      }      else if (digitalRead(del) == 0)      {        id = count;        deleteFingerprint(id);        return;      }      else if (digitalRead(enroll) == 0)      {        return;      }    }  }    uint8\_t getFingerprintEnroll()  {    int p = -1;    lcd.clear();    lcd.print("finger ID:");    lcd.print(id);    lcd.setCursor(0, 1);    lcd.print("Place Finger");    delay(2000);      while (p != FINGERPRINT\_OK)    {      p = finger.getImage();      switch (p)      {        case FINGERPRINT\_OK:          Serial.println("Image taken");          lcd.clear();          lcd.print("Image taken");          break;        case FINGERPRINT\_NOFINGER:          Serial.println("No Finger");          lcd.clear();          lcd.print("No Finger");          break;        case FINGERPRINT\_PACKETRECIEVEERR:          Serial.println("Communication error");          lcd.clear();          lcd.print("Comm Error");          break;        case FINGERPRINT\_IMAGEFAIL:          Serial.println("Imaging error");          lcd.clear();          lcd.print("Imaging Error");          break;        default:          Serial.println("Unknown error");          lcd.clear();          lcd.print("Unknown Error");          break;      }    }      p = finger.image2Tz(1);    switch (p) {      case FINGERPRINT\_OK:        Serial.println("Image converted");        lcd.clear();        lcd.print("Image converted");        break;      case FINGERPRINT\_IMAGEMESS:        Serial.println("Image too messy");        lcd.clear();        lcd.print("Image too messy");        return p;      case FINGERPRINT\_PACKETRECIEVEERR:        Serial.println("Communication error");        lcd.clear();        lcd.print("Comm Error");        return p;      case FINGERPRINT\_FEATUREFAIL:        Serial.println("Could not find fingerprint features");        lcd.clear();        lcd.print("Feature Not Found");        return p;      case FINGERPRINT\_INVALIDIMAGE:        Serial.println("Could not find fingerprint features");        lcd.clear();        lcd.print("Feature Not Found");        return p;      default:        Serial.println("Unknown error");        lcd.clear();        lcd.print("Unknown Error");        return p;    }      Serial.println("Remove finger");    lcd.clear();    lcd.print("Remove Finger");    delay(2000);    p = 0;      while (p != FINGERPRINT\_NOFINGER) {      p = finger.getImage();    }      Serial.print("ID "); Serial.println(id);    p = -1;    Serial.println("Place same finger again");    lcd.clear();    lcd.print("Place Finger");    lcd.setCursor(0, 1);    lcd.print("   Again");      while (p != FINGERPRINT\_OK) {      p = finger.getImage();      switch (p) {        case FINGERPRINT\_OK:          Serial.println("Image taken");          break;        case FINGERPRINT\_NOFINGER:          Serial.print(".");          break;        case FINGERPRINT\_PACKETRECIEVEERR:          Serial.println("Communication error");          break;        case FINGERPRINT\_IMAGEFAIL:          Serial.println("Imaging error");          break;        default:          Serial.println("Unknown error");          return;      }    }      p = finger.image2Tz(2);      switch (p) {      case FINGERPRINT\_OK:        Serial.println("Image converted");        break;      case FINGERPRINT\_IMAGEMESS:        Serial.println("Image too messy");        return p;      case FINGERPRINT\_PACKETRECIEVEERR:        Serial.println("Communication error");        return p;      case FINGERPRINT\_FEATUREFAIL:        Serial.println("Could not find fingerprint features");        return p;      case FINGERPRINT\_INVALIDIMAGE:        Serial.println("Could not find fingerprint features");        return p;      default:        Serial.println("Unknown error");        return p;    }      Serial.print("Creating model for #");  Serial.println(id);      p = finger.createModel();    if (p == FINGERPRINT\_OK) {      Serial.println("Prints matched!");    } else if (p == FINGERPRINT\_PACKETRECIEVEERR) {      Serial.println("Communication error");      return p;    } else if (p == FINGERPRINT\_ENROLLMISMATCH) {      Serial.println("Fingerprints did not match");      return p;    } else {      Serial.println("Unknown error");      return p;    }      Serial.print("ID "); Serial.println(id);    p = finger.storeModel(id);    if (p == FINGERPRINT\_OK) {      Serial.println("Stored!");      lcd.clear();      lcd.print("Stored!");      delay(2000);    } else if (p == FINGERPRINT\_PACKETRECIEVEERR) {      Serial.println("Communication error");      return p;    } else if (p == FINGERPRINT\_BADLOCATION) {      Serial.println("Could not store in that location");      return p;    } else if (p == FINGERPRINT\_FLASHERR) {      Serial.println("Error writing to flash");      return p;    } else {      Serial.println("Unknown error");      return p;    }  }    int getFingerprintIDez()  {    uint8\_t p = finger.getImage();    if (p != FINGERPRINT\_OK)      return -1;    p = finger.image2Tz();    if (p != FINGERPRINT\_OK)      return -1;      p = finger.fingerFastSearch();    if (p != FINGERPRINT\_OK)    {      lcd.clear();      lcd.print("Finger Not Found");      lcd.setCursor(0, 1);      lcd.print("Try Later");      delay(2000);      return -1;    }    // found a match!    Serial.print("Found ID #");    Serial.print(finger.fingerID);    return finger.fingerID;  }    uint8\_t deleteFingerprint(uint8\_t id)  {    uint8\_t p = -1;    lcd.clear();    lcd.print("Please wait");    p = finger.deleteModel(id);      if (p == FINGERPRINT\_OK)    {      Serial.println("Deleted!");      lcd.clear();      lcd.print("Figer Deleted");      lcd.setCursor(0, 1);      lcd.print("Successfully");      delay(1000);    }    else    {      Serial.print("Something Wrong");      lcd.clear();      lcd.print("Something Wrong");      lcd.setCursor(0, 1);      lcd.print("Try Again Later");      delay(2000);      return p;    }  } |

Đảm bảo bo mạch Arduino của bạn được chọn trong “ **Tools** “ > **“Board”** và chọn đúng cổng trong **“Tools”** > **“Port”.** Nhấp vào nút **“Tải lên”** để tải mã lên Arduino.

### Hệ thống bảo mật sinh trắc học Arduino hoạt động như thế nào

Sau khi tải đoạn mã trên lên Arduino Nano Board, dự án hệ thống đã sẵn sàng để thử nghiệm.

[](https://diyprojectslab.com/wp-content/uploads/First-1-scaled.jpg)

Đó là nó! Hãy làm theo các bước sau để sử dụng hệ thống nhằm kiểm soát truy cập an toàn bằng nhận dạng vân tay.

* **Đăng ký/Quay lại :** Đăng ký vân tay mới hoặc quay lại.
* **Xóa/OK :** Xóa dấu vân tay hoặc xác nhận hành động.
* **Lên/Khớp :** Di chuyển lên hoặc tăng các giá trị và bắt đầu xác minh dấu vân tay.
* **Xuống/Khớp :** Di chuyển xuống hoặc giảm các giá trị và sử dụng để xác minh dấu vân tay.

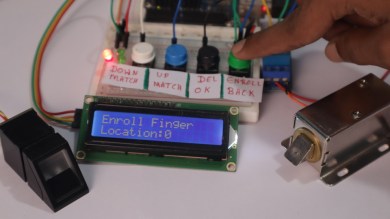
##### ****1. Bật nguồn:****

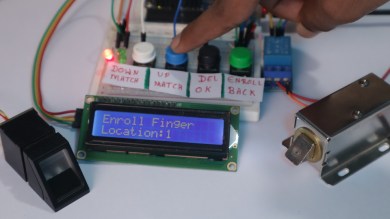
* Cung cấp cho hệ thống nguồn điện bằng nguồn điện **12V DC** .

##### ****2. Bắt đầu thiết lập:****

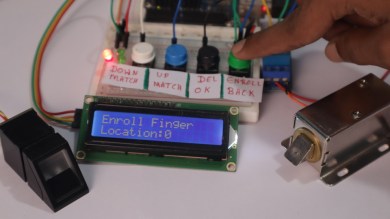
* + Hệ thống sẽ khởi động và hiển thị “ **Hệ thống bảo mật vân tay** ”.
  + Chờ người dùng nhập liệu.

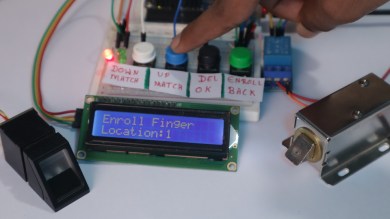
##### ****3. Đăng ký dấu vân tay:****

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Enroll-Button.jpg?ssl=1)

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Up-Enroll_1.jpg?ssl=1)

* Nhấn nút “ **Đăng ký** ” .
* Màn hình LCD sẽ hiển thị cho bạn thông qua thủ tục đăng ký:
  + Chọn vị trí cho dấu vân tay duy nhất bằng nút Lên và Xuống ( **1 đến 25** ).
  + Đặt ngón tay của bạn lên cảm biến theo yêu cầu.
  + Lặp lại quy trình cho đến khi đăng ký xong.
* Hệ thống lưu trữ dữ liệu dấu vân tay.

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Enroll-Button.jpg?ssl=1)

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Up-Enroll_1.jpg?ssl=1)

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Biometric-Stored-Sucess.jpg?ssl=1)

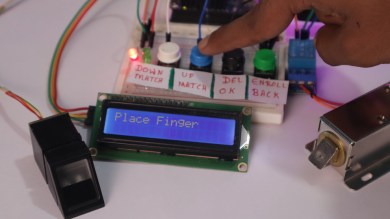
##### ****4. Xóa dấu vân tay:****

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Select-Delete-Finger.jpg?ssl=1)

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Finger-Deleted.jpg?ssl=1)

* Nhấn “ **Xóa** ” để xóa dấu vân tay.
* Làm theo Màn hình LCD để chọn cái cần xóa.

##### ****5. Cấp quyền truy cập:****

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Push-Up-or-DOwn-Switch.jpg?ssl=1)

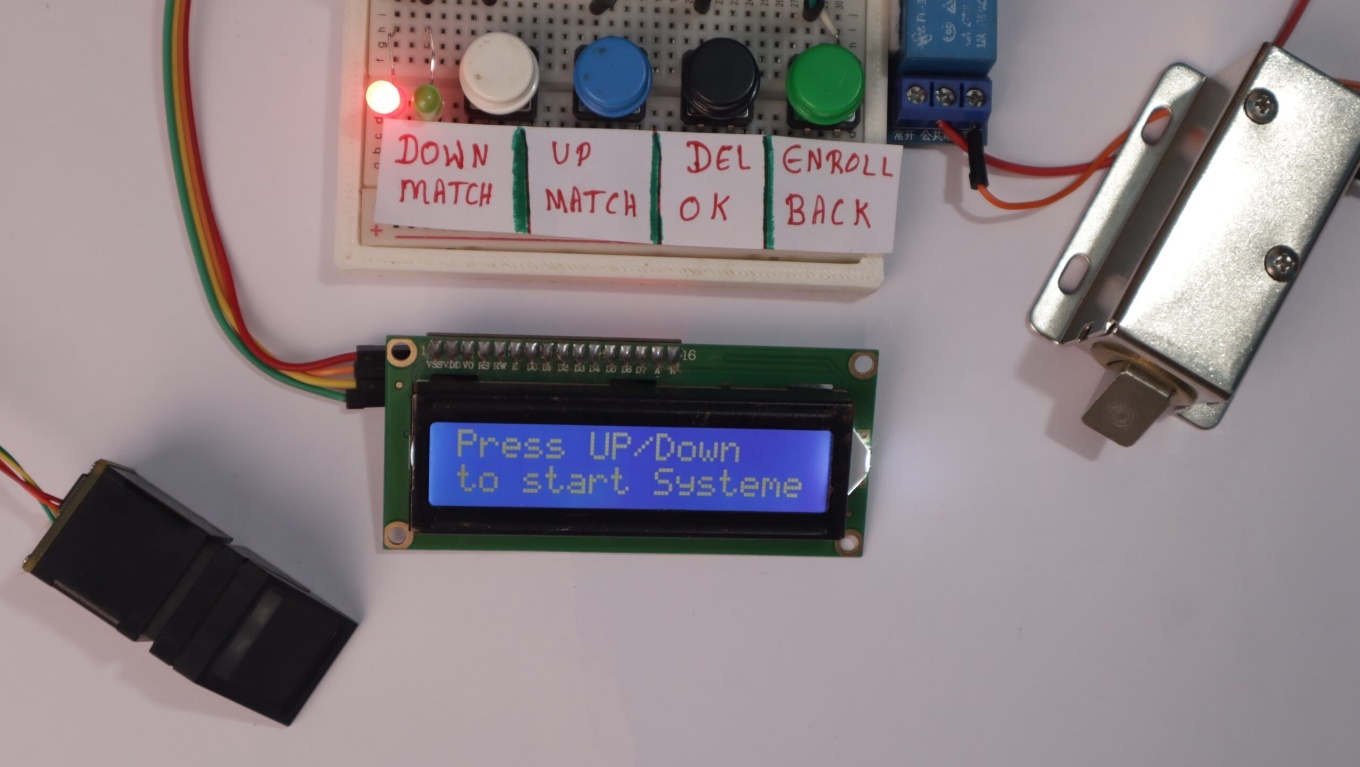
[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/Place-Right-Finger.jpg?ssl=1)

* Nhấn “ **Lên** ” hoặc “ **Xuống** ” để truy cập.
* Đặt ngón tay lên cảm biến khi màn hình LCD hiển thị cho bạn.
* Nếu được xác định, cửa sẽ mở khóa (rơ-le kích hoạt).
* Màn hình LCD hiển thị “ **Cổng đã mở** ” . “

##### ****6. Hết thời gian truy cập:****

* Sau một hồi trì hoãn, cửa khóa lại.
* Màn hình LCD hiển thị “ **Cổng đã đóng”.** “

## Phần kết luận

[](https://diyprojectslab.com/wp-content/uploads/First-1-scaled.jpg)

Hệ thống bảo mật sinh trắc học, với Arduino và cảm biến vân tay, tận dụng khả năng của những tiến bộ đáng kể trong các giải pháp bảo mật sử dụng dấu vân tay. Tóm lại, dự án này trình bày cách xây dựng Hệ thống bảo mật sinh trắc học.

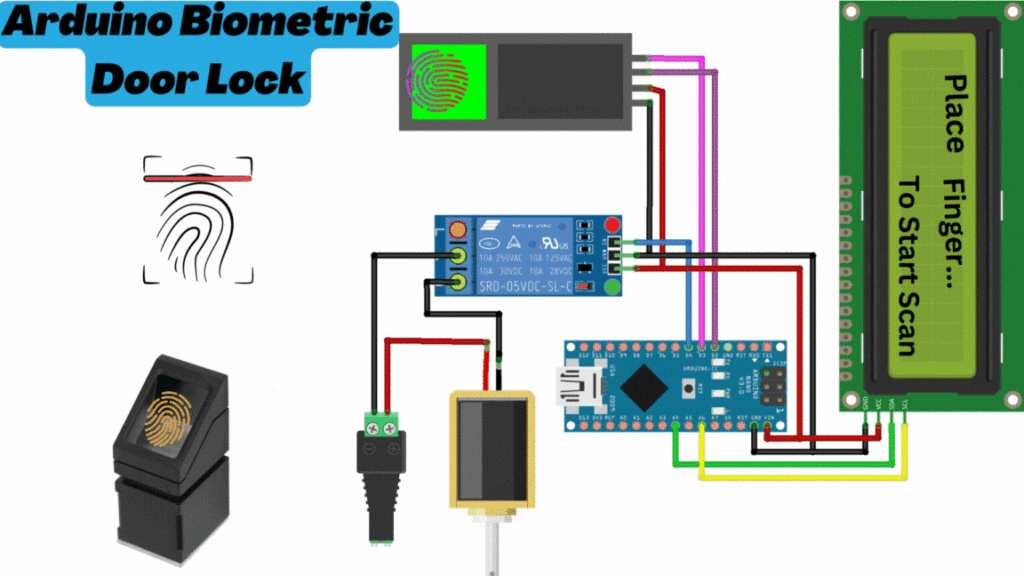
# Fingerprint Door Lock Security Systems Using Arduino & LCD

<https://diyprojectslab.com/fingerprint-door-lock-security-arduino/>

# Hệ Thống Bảo Mật Khóa Cửa Vân Tay Sử Dụng Arduino & LCD

Quản trị viênBỞI [QUẢN TRỊ VIÊN](https://diyprojectslab.com/author/diyprojectslab/)[MIỄN BÌNH LUẬN](https://diyprojectslab.com/fingerprint-door-lock-security-arduino/#respond)3 PHÚT ĐỌC

**Chia sẻ**



**CHIA SẺ**

Trong dự án này, chúng tôi sẽ xây dựng một hệ thống bảo mật khóa cửa vân tay bằng Arduino. Hệ thống này có thể được sử dụng trong nhà, văn phòng, v.v. của chúng ta để đảm bảo an ninh. Chúng ta cũng có thể sử dụng tính năng này để mở cửa bằng cách chỉ cần đặt ngón tay lên ổ khóa.

An ninh là ưu tiên hàng đầu của nhiều chủ nhà. Hệ thống bảo mật khóa cửa vân tay là một cách tuyệt vời để thêm một lớp bảo mật bổ sung cho ngôi nhà của bạn. hệ thống khóa cửa này sử dụng [đầu đọc dấu vân tay](https://en.wikipedia.org/wiki/Fingerprint_scanner) để mở khóa cửa. đầu đọc dấu vân tay được kết nối với bo mạch vi điều khiển Arduino.

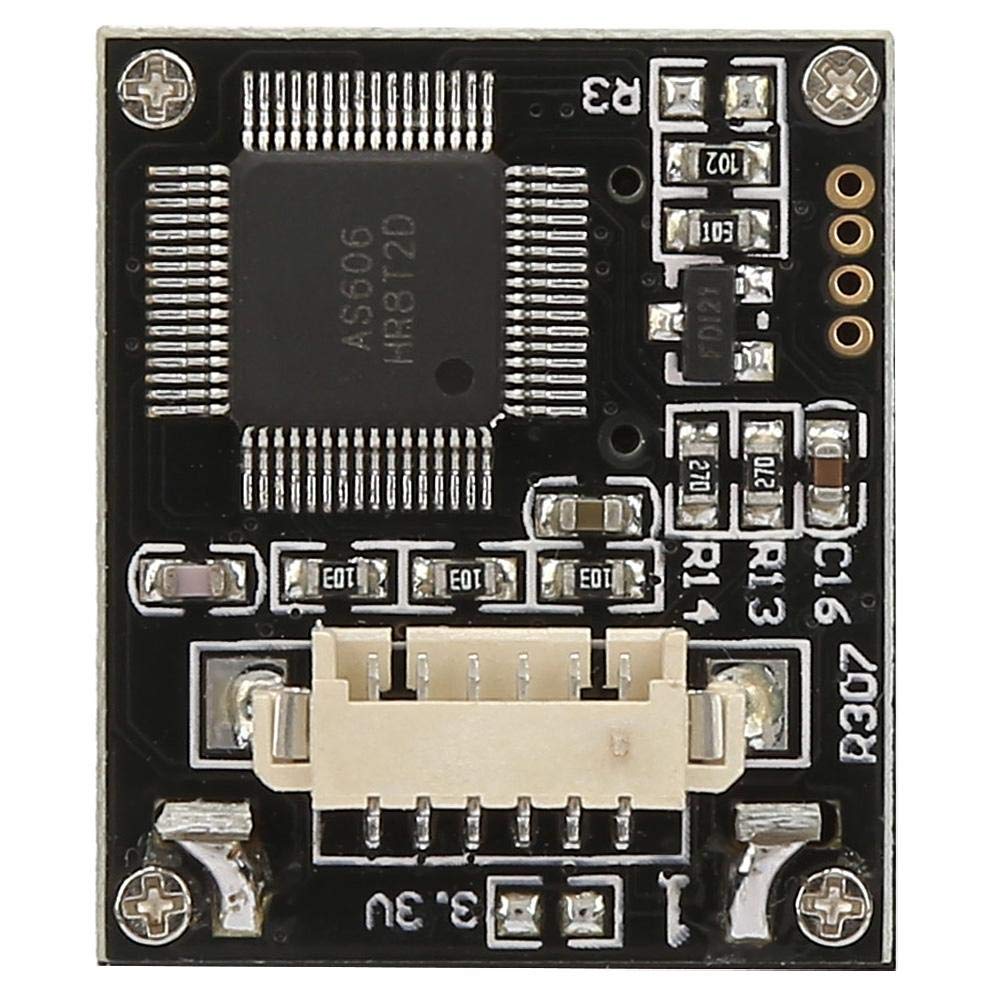
### Vật liệu cần thiết

* bo mạch Arduino
* Mô-đun cảm biến vân tay R307
* Mô-đun LCD I2C 16 × 2
* Khóa điện từ
* Dây nhảy
* Breadboard
* Pin 9V (tùy chọn)
* cáp USB

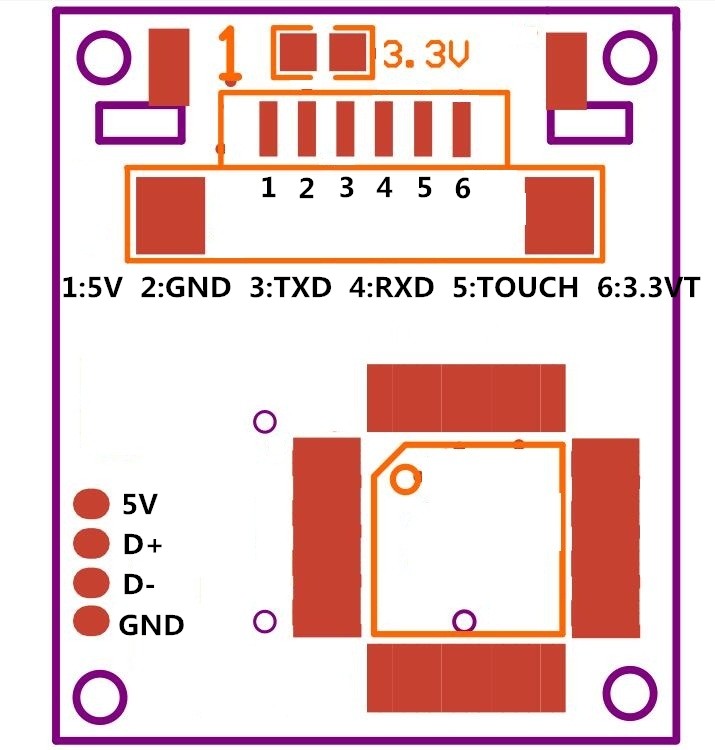
### Mô-đun cảm biến vân tay R307

[](https://diyprojectslab.com/wp-content/uploads/2023/04/R307-Optical-Fingerprint-Reader-Sensor-Module.jpg)

Mô-đun cảm biến vân tay R307 là mô-đun đọc dấu vân tay sinh trắc học chi phí thấp, có thể dễ dàng giao tiếp với các bộ vi điều khiển như Arduino/Raspberry Pi.

Với thuật toán nhận dạng dấu vân tay tích hợp, nó cũng có thể được lập trình dễ dàng chỉ với một vài dòng mã để khớp, xác minh, tìm kiếm hoặc lưu trữ dấu vân tay. Hãy xem bài đăng cuối cùng của chúng tôi – [**Giao tiếp cảm biến vân tay R307 với Arduino**](https://diyprojectslab.com/r307-fingerprint-sensor-with-arduino/)[](https://diyprojectslab.com/wp-content/uploads/2023/04/R307-Optical-Fingerprint-Sensor-Module-back.jpg)

#### Mô-đun cảm biến vân tay R307 Pinout

[](https://diyprojectslab.com/wp-content/uploads/2023/04/Pinout-R307-Fingerprint-Sensor.jpg)

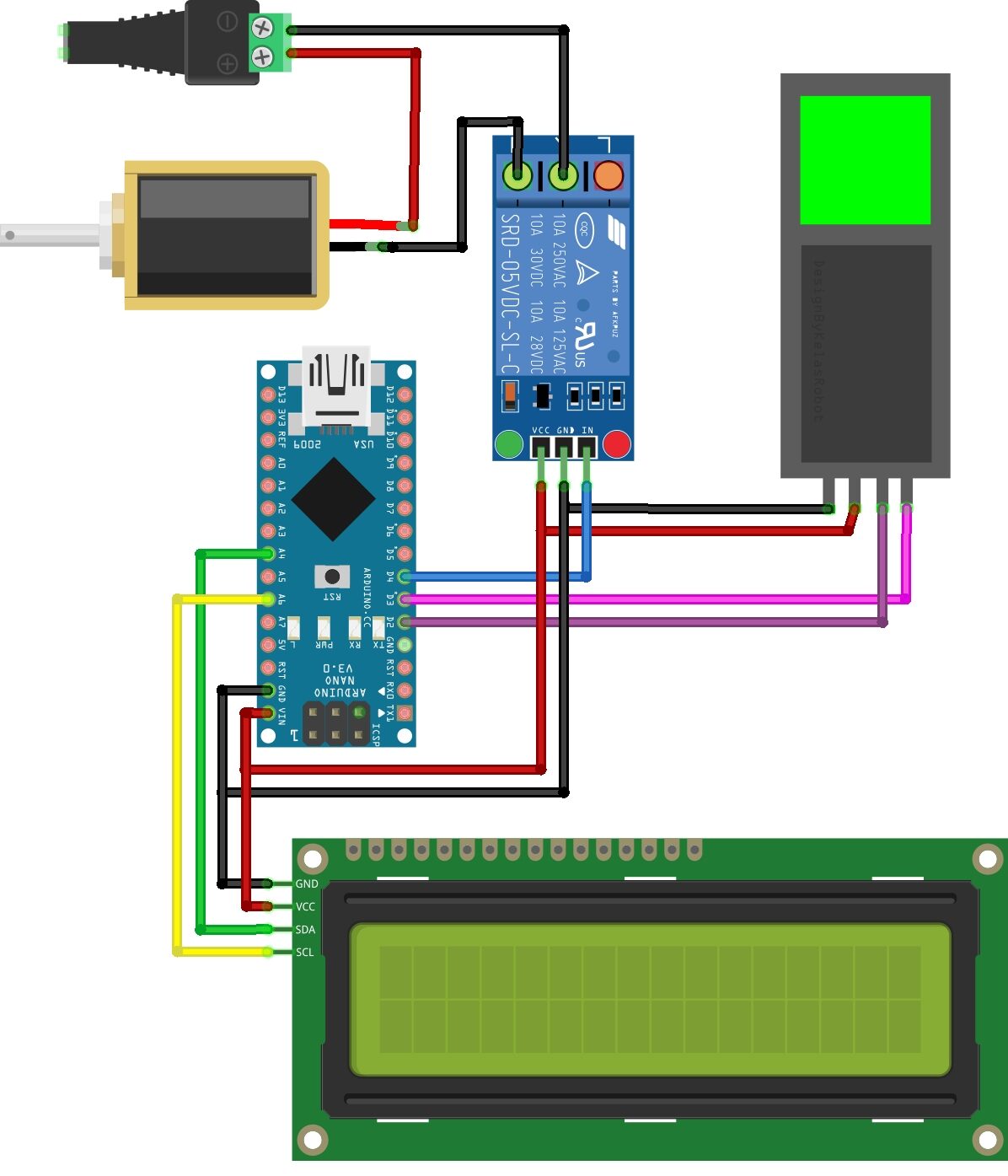
#### Thông số kỹ thuật mô-đun cảm biến vân tay R307

**Dưới đây là một số thông số kỹ thuật của cảm biến vân tay R307**

* **Giao diện** : UART(TTL)
* **Điện áp hoạt động** : 3.6V – 6.0V DC
* **Độ phân giải hình ảnh vân tay** : 500dpi
* **Tỷ lệ chấp nhận sai (FAR):** <0,001% (mức bảo mật 3)
* **Tỷ lệ từ chối sai (FRR)** : <1,0% (mức bảo mật 3)
* **Tốc độ nhận dạng** : <0,5s
* **Dung lượng lưu trữ mẫu** : lên tới 1000 mẫu
* **Nhiệt độ hoạt động** : -20°C đến +50°C
* **Độ ẩm hoạt động** : 40% RH – 85% RH
* **Kích thước** : 54mm x 20mm x 22mm

### Sơ đồ – Hệ thống bảo mật khóa cửa vân tay sử dụng Arduino & LCD

Kết nối cảm biến vân tay với bo mạch Arduino như sau:

[](https://diyprojectslab.com/wp-content/uploads/2023/04/Fingerprint-Door-Lock-Security-Systems-Using-Arduino-Uno-lcd.jpg)**Sơ đồ mạch**

* Chân VCC đến chân 5V trên Arduino
* Chân GND tới chân GND trên Arduino
* Chân TX sang chân kỹ thuật số D2 trên Arduino
* Chân RX đến chân kỹ thuật số D3 trên Arduino
* Chân Rơle D4
* LCD SDA -> Chân tương tự Arduino 4
* LCD SCL -> Chân tương tự Arduino 5
* LCD VCC -> Arduino 5V
* LCD GND -> Arduino GND

## Mã & thư viện

Trước tiên, hãy tải xuống thư viện Vân tay Adafruit từ Trình quản lý thư viện Arduino IDE. hãy xem hướng dẫn này [**Cảm biến vân tay với Arduino**](https://diyprojectslab.com/r307-fingerprint-sensor-with-arduino/)

Mở Arduino IDE, đi tới Tệp > Ví dụ > Vân tay Adafruit > đăng ký. và Tải mã lên Arduino của bạn.

### Mã Đăng Ký – Cảm Biến Vân Tay R307

Đây là mã mẫu:

#include <Adafruit\_Fingerprint.h>

#if (defined(\_\_AVR\_\_) || defined(ESP8266)) && !defined(\_\_AVR\_ATmega2560\_\_)

// For UNO and others without hardware serial, we must use software serial...

// pin #2 is IN from sensor (GREEN wire)

// pin #3 is OUT from arduino (WHITE wire)

// Set up the serial port to use softwareserial..

SoftwareSerial mySerial(2, 3);

#else

// On Leonardo/M0/etc, others with hardware serial, use hardware serial!

// #0 is green wire, #1 is white

#define mySerial Serial1

#endif

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

uint8\_t id;

void setup()

{

Serial.begin(9600);

while (!Serial); // For Yun/Leo/Micro/Zero/...

delay(100);

Serial.println("\n\nAdafruit Fingerprint sensor enrollment");

// set the data rate for the sensor serial port

finger.begin(57600);

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1) { delay(1); }

}

Serial.println(F("Reading sensor parameters"));

finger.getParameters();

Serial.print(F("Status: 0x")); Serial.println(finger.status\_reg, HEX);

Serial.print(F("Sys ID: 0x")); Serial.println(finger.system\_id, HEX);

Serial.print(F("Capacity: ")); Serial.println(finger.capacity);

Serial.print(F("Security level: ")); Serial.println(finger.security\_level);

Serial.print(F("Device address: ")); Serial.println(finger.device\_addr, HEX);

Serial.print(F("Packet len: ")); Serial.println(finger.packet\_len);

Serial.print(F("Baud rate: ")); Serial.println(finger.baud\_rate);

}

uint8\_t readnumber(void) {

uint8\_t num = 0;

while (num == 0) {

while (! Serial.available());

num = Serial.parseInt();

}

return num;

}

void loop() // run over and over again

{

Serial.println("Ready to enroll a fingerprint!");

Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");

id = readnumber();

if (id == 0) {// ID #0 not allowed, try again!

return;

}

Serial.print("Enrolling ID #");

Serial.println(id);

while (! getFingerprintEnroll() );

}

uint8\_t getFingerprintEnroll() {

int p = -1;

Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

Serial.println("Remove finger");

delay(2000);

p = 0;

while (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

Serial.print("ID "); Serial.println(id);

p = -1;

Serial.println("Place same finger again");

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.print(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(2);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK converted!

Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();

if (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_ENROLLMISMATCH) {

Serial.println("Fingerprints did not match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

Serial.print("ID "); Serial.println(id);

p = finger.storeModel(id);

if (p == FINGERPRINT\_OK) {

Serial.println("Stored!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_BADLOCATION) {

Serial.println("Could not store in that location");

return p;

} else if (p == FINGERPRINT\_FLASHERR) {

Serial.println("Error writing to flash");

return p;

} else {

Serial.println("Unknown error");

return p;

}

return true;

}

Mở Serial Monitor để xem tin nhắn.

[](https://diyprojectslab.com/wp-content/uploads/2023/04/Enroll-Fingerprint-3.jpg)

### Mã nguồn hệ thống khóa cửa vân tay

#include <Adafruit\_Fingerprint.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

SoftwareSerial mySerial(2, 3); //Serial Communications

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

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

#define Relay 4

void setup()

{

lcd.init();

lcd.backlight();

finger.begin(57600);

pinMode(Relay, OUTPUT);

digitalWrite(Relay, LOW);

delay(5);

if (finger.verifyPassword()) {

lcd.setCursor(0,0);

lcd.print(" Finger Print ");

lcd.setCursor(0, 1);

lcd.print("Sensor Connected");

delay (3000);

} else {

lcd.setCursor(0,0);

lcd.print("Unable to found ");

lcd.setCursor(0,1);

lcd.print("Sensor");

delay(2000);

lcd.setCursor(0, 0);

lcd.print("Please Check ");

lcd.setCursor(0,1);

lcd.print(" Sensor Wiring ");

while (1) {

delay(1); }

}

}

void loop()

{

lcd.setCursor(0,0);

lcd.print("Place finger...");

lcd.setCursor(0,1);

lcd.print(" start scan ");

getFingerprintID();

delay(50); //don't ned to run this at full speed.

}

uint8\_t getFingerprintID() {

uint8\_t p = finger.getImage();

if(p == FINGERPRINT\_NOFINGER){

return p;

}

else if(p != FINGERPRINT\_OK){

lcd.setCursor(0,0);

lcd.print("Scan Error ");

lcd.setCursor(0,1);

delay(2000);

return p;

}

p = finger.image2Tz();

if(p != FINGERPRINT\_OK){

lcd.setCursor(0,0);

lcd.print("Processing Error");

lcd.setCursor(0,1);

lcd.print(" Try Again ");

return p;

}

p = finger.fingerSearch();

if (p == FINGERPRINT\_OK) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(" Door Unlocked");

lcd.setCursor(0, 1);

lcd.print(" Welcome");

digitalWrite(Relay,HIGH);

delay(5000);

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

lcd.setCursor(0,0);

lcd.print("Comm Error ");

lcd.setCursor(0,1);

lcd.print(" ");

delay(2000);

return p;

} else if (p == FINGERPRINT\_NOTFOUND) {

lcd.setCursor(0,0);

lcd.print("Access Denied ");

lcd.setCursor(0,1);

lcd.print(" ");

delay(2000);

return p;

} else {

lcd.setCursor(0,0);

lcd.print("Error in matching");

lcd.setCursor(0,1);

lcd.print("Not Valid Finger");

delay(2000);

return p;

}

digitalWrite(Relay,LOW);

return finger.fingerID;

}

**Phần kết luận**

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/2023/04/fingerprint-door-lock-system-scaled.jpeg?ssl=1)

[](https://i0.wp.com/diyprojectslab.com/wp-content/uploads/2023/04/Arduino-fingerprint-door-lock-system.jpeg?ssl=1)

<https://iotprojectsideas.com/fingerprint-door-lock-security-systems-using-arduino-lcd/>