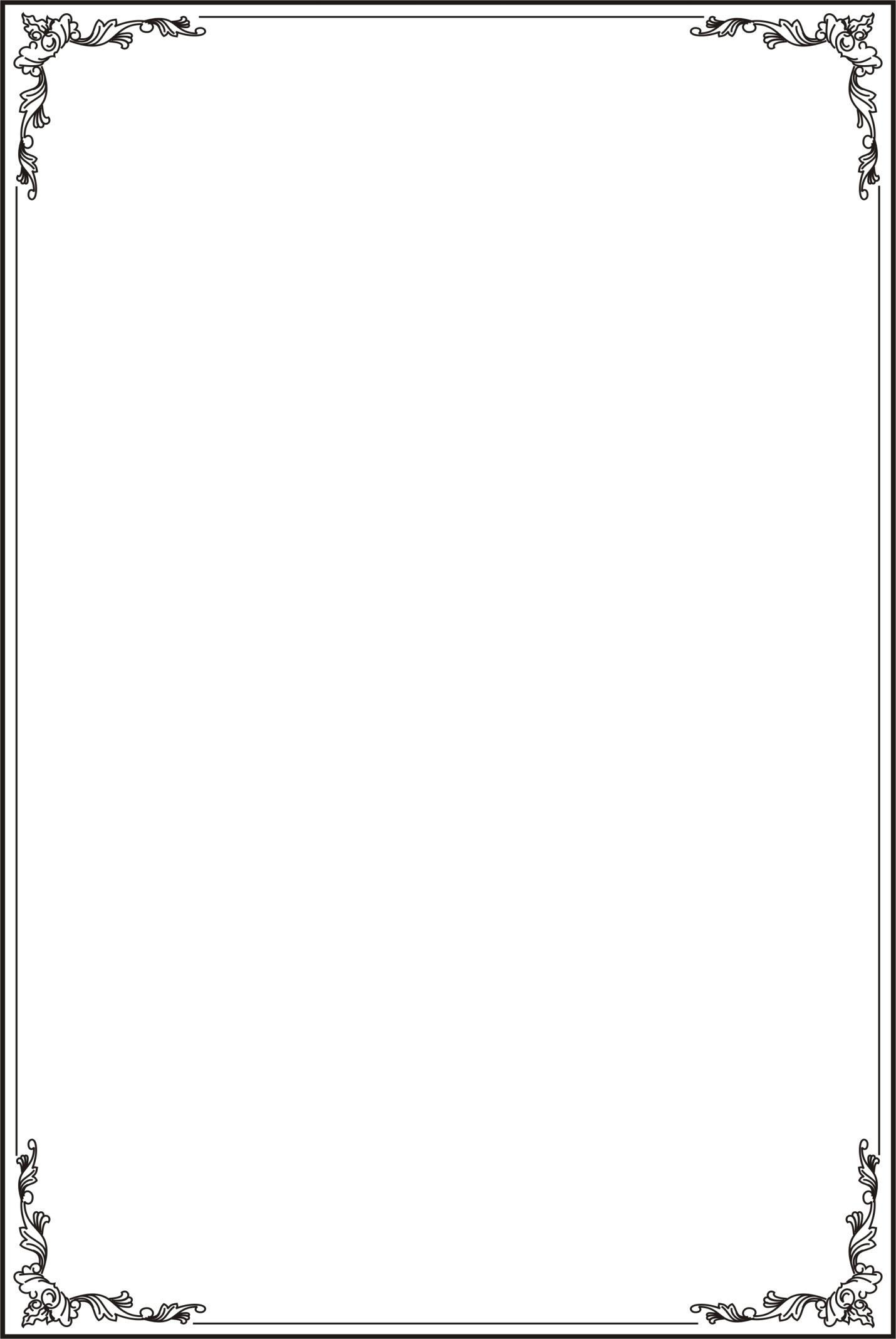
**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CƠ CÔNG NGHỆ THÔNG TIN**



****

**BÁO CÁO KẾT THÚC HỌC PHẦN IOT VÀ ỨNG DỤNG**

**Nhóm 14 : HỆ THỐNG KHÓA CỬA THÔNG MINH**

**Giảng viên: Nguyễn Tài Tuyên**

**Thành viên: Lê Đình Dương**

**Tiêu Hoàng Đạt**

***Hà Nội – 2024***

# MỤC LỤC

[**MỤC LỤC 1**](#_heading=h.dp27otvo7r0x)

[**DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT 2**](#_heading=h.orziga6qnovj)

[1.1:Giới thiệu đề tài 4](#_heading=h.30j0zll)

[1.2.Các tính năng chính của hệ thống 4](#_heading=)

[**Chương 2 Hệ thống phần cứng 7**](#_heading=h.1fob9te)

[2.1:Tổng quan các module được sử dụng trong dự án 7](#_heading=h.3znysh7)

[2.2 Chi tiết về các module và chức năng 9](#_heading=h.2et92p0)

[2.2.1.ESP32 9](#_heading=h.tyjcwt)

[2.2.2.Bàn phím 13](#_heading=h.gsstmgaxdoct)

[2.2.3.Màn hình Led 14](#_heading=h.4yy1at1kgrh1)

[2.2.4.Còi 16](#_heading=h.lho3998bmby0)

[2.2.5. Servo 17](#_heading=h.q125ylkp14bf)

[2.2.6. Công tắc giới hạn (Limit Switch) 20](#_heading=h.p4ks0hyd564h)

[2.2.7. ESP32 CAM 21](#_heading=h.d314s7z0y57e)

[2.3 Mô tả về cách hoạt động và mã nguồn Arduino 23](#_heading=h.35nkun2)

[2.3.1. Sơ đồ mạch tổng quan 23](#_heading=h.1ksv4uv)

[2.3.2.Mô tả nguyên tắc hoạt động 24](#_heading=h.44sinio)

[2.3.3. Mã nguồn Aduino 25](#_heading=h.zhxp4gbote4a)

[2.4 Giải thích tổng quan về mã nguồn 38](#_heading=h.3j2qqm3)

[Chương 3 Hệ thống phần mềm 40](#_heading=h.4i7ojhp)

[3.1 Phiên bản app trên điện thoại 40](#_heading=h.7ek0ob2t23f8)

[3.1.1 Mục đích 40](#_heading=h.kpnw7jxw5f29)

[3.1.2 Công nghệ sử dụng 40](#_heading=h.ak9rwbdb9ppm)

[3.1.3 Các tính năng 41](#_heading=h.etpjk5r8nqeu)

[3.2 Phiên bản web 47](#_heading=h.eqszcoy8elg5)

[3.2.1 Mục đích 47](#_heading=h.u6sppuradvpq)

[3.2.2 Công nghệ sử dụng 47](#_heading=h.s3q3h344oqbf)

[3.2.3 Các tính năng 48](#_heading=h.jq4u5ym43bi5)

[**Chương 4 Kết quả 57**](#_heading=h.147n2zr)

[4.1. Chức năng đã hoàn thành 57](#_heading=h.3o7alnk)

[4.2. Hiệu suất hệ thống 57](#_heading=h.23ckvvd)

[4.3. Cải tiến trong tương lai 58](#_heading=h.ihv636)

[**Lời cảm ơn 59**](#_heading=h.32hioqz)

# 

# DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT

| Viết tắt | Tiếng Anh | Tiếng Việt |
| --- | --- | --- |
| IOT | Internet of Things | Internet vạn vật |
| API | Application Programming Interface | Giao diện lập trình ứng dụng |
| URL | Uniform Resource Locator | Địa chỉ dẫn tới tài nguyên |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| GPIO | General Purpose Input Output | Cổng vào/ra đa dụng |
| LED | Light Emitting Diode | Điốt phát sáng |
| HTTP | HyperText Transfer Protocol | Giao thức truyền tải siêu văn bản |
| SPI | Serial Peripheral Interface | Giao tiếp ngoại vi nối tiếp |
| I2C | Inter-Integrated Circuit | Giao tiếp liên tích hợp |
| UART | Universal Asynchronous Receiver-Transmitter | Bộ thu-phát bất đồng bộ đa năng |
| I2S | Inter-IC Sound | Giao tiếp âm thanh giữa các vi mạch |
| CAN | Controller Area Network | Mạng vùng điều khiển |
| PWM | Pulse Width Modulation | Điều chỉnh độ rộng xung |
| OLED | Organic Light Emitting Diode | Điốt phát quang hữu cơ |
| IP | Internet Protocol | Giao thức internet |
| PWM | Pulse Width Modulation | Điều chỉnh độ rộng xung |
| LCD | Liquid Crystal Display | Màn hình tinh thể lỏng |
| APK | Android Package | Gói cài đặt android |

Chương 1 Giới thiệu chung

## 1.1:Giới thiệu đề tài

Khóa cửa thông minh là một trong những ứng dụng nổi bật của công nghệ Internet of Things (IoT) trong lĩnh vực bảo mật và quản lý quyền truy cập. Khác với các hệ thống khóa cửa truyền thống, khóa cửa thông minh sử dụng các phương thức nhận diện hiện đại như mã số, vân tay, hoặc kết nối với các thiết bị di động để cung cấp sự tiện lợi và an toàn tối ưu cho người dùng. Đặc biệt, với khả năng điều khiển từ xa và tự động hóa, khóa cửa thông minh mang lại nhiều lợi ích vượt trội trong việc bảo vệ tài sản và giám sát quyền truy cập.

Một trong những điểm mạnh của khóa cửa thông minh là tính năng điều khiển từ xa qua các ứng dụng di động. Người dùng có thể dễ dàng mở hoặc khóa cửa từ bất kỳ đâu thông qua điện thoại của mình, mà không cần phải có chìa khóa vật lý. Bên cạnh đó, công nghệ nhận diện sinh trắc học như vân tay hay mã số giúp tăng cường tính bảo mật, chỉ cho phép những người có quyền truy cập hợp lệ mới có thể mở cửa.

Khóa cửa thông minh không chỉ mang lại sự tiện ích cho các hộ gia đình mà còn được ứng dụng rộng rãi trong các văn phòng, khách sạn hay các tòa nhà cao tầng. Nhờ vào khả năng ghi lại lịch sử truy cập, người quản lý có thể dễ dàng theo dõi việc ra vào của các thành viên hay khách mời, đồng thời giúp kiểm soát và hạn chế những truy cập trái phép.

Với sự phát triển không ngừng của các công nghệ mới, khóa cửa thông minh sẽ ngày càng trở nên phổ biến và là lựa chọn hàng đầu cho các giải pháp bảo mật trong tương lai. Việc nghiên cứu và phát triển các hệ thống khóa cửa thông minh sẽ mở ra cơ hội mới trong việc nâng cao chất lượng cuộc sống, đồng thời cải thiện hiệu quả công tác bảo mật tại nhiều môi trường khác nhau.

## 1.2.Các tính năng chính của hệ thống

* **Điều khiển từ xa:** Khóa cửa thông minh cho phép người dùng mở hoặc khóa cửa từ xa thông qua các ứng dụng di động hoặc các thiết bị điều khiển từ xa với các thiết bị kết nối Internet. Điều này rất hữu ích khi bạn quên mang chìa khóa hoặc muốn cho phép ai đó vào nhà trong khi bạn không có mặt. Ví dụ, nếu bạn có khách đến nhà khi vắng mặt, bạn có thể mở cửa từ xa qua điện thoại mà không cần phải trao chìa khóa vật lý. Điều này cũng giúp cải thiện sự tiện nghi trong các trường hợp khẩn cấp, như khi người thân hoặc bạn bè cần vào nhà.
* **Mã số:** Một trong những tính năng nổi bật của khóa cửa thông minh là khả năng sử dụng phương thức nhận dạng mã số thay thế chìa khóa vật lý. Khóa này có thể được mở bằng cách nhập mã số. Phương thức này mang lại mức độ bảo mật cao hơn so với việc sử dụng chìa khóa truyền thống, vì chúng khó bị sao chép hay đánh cắp. Việc nhận dạng vân tay là rất phổ biến vì nó nhanh chóng, chính xác và không yêu cầu người dùng nhớ mã số hoặc mang theo chìa khóa.Mã số sẽ được quản lí ở trên internet thông qua ứng dụng nên rất an toàn.
* **Kết nối với ứng dụng di động:** Khóa cửa thông minh thường đi kèm với ứng dụng di động được cài đặt trên điện thoại thông minh của người dùng. Ứng dụng này không chỉ giúp mở hoặc khóa cửa từ xa mà còn cung cấp các thông tin quan trọng về trạng thái khóa cửa. Người dùng có thể nhận thông báo khi cửa được mở hoặc đóng, hoặc khi có ai đó cố gắng mở cửa. Một số hệ thống còn cung cấp thông báo về các hoạt động bất thường, như khi có ai đó thử mở cửa mà không có quyền truy cập. Điều này không chỉ giúp tăng cường bảo mật mà còn cung cấp sự kiểm soát hoàn toàn đối với cửa ra vào.
* **Lịch sử truy cập:** Lịch sử truy cập là một tính năng quan trọng giúp người quản lý hoặc chủ nhà theo dõi các hoạt động ra vào cửa. Khóa cửa thông minh lưu trữ thông tin về thời gian và người đã mở cửa, giúp dễ dàng kiểm soát việc ra vào của các thành viên trong gia đình, bạn bè, hoặc khách mời. Chức năng này có thể cung cấp dữ liệu chi tiết về các lần truy cập, giúp chủ nhà xác định được ai đã ra vào nhà và vào lúc nào, từ đó nâng cao tính an toàn và bảo mật. Ngoài ra, nó cũng hữu ích trong việc quản lý quyền truy cập của các nhân viên hoặc khách mời trong các môi trường văn phòng hoặc khách sạn.
* **Khóa cửa tự động:** Khóa cửa tự động là một tính năng tiện ích trong các hệ thống khóa cửa thông minh. Khi cửa được đóng lại, khóa sẽ tự động kích hoạt để đảm bảo rằng cửa được khóa mà không cần người dùng phải thao tác thủ công. Điều này rất hữu ích trong các trường hợp mà người dùng quên khóa cửa khi ra ngoài. Ngoài ra, một số hệ thống còn có thể mở cửa tự động khi nhận diện tín hiệu từ thiết bị di động của người dùng, như khi người dùng đến gần cửa với điện thoại của mình. Tính năng này giúp tăng sự tiện lợi và an toàn, vì người dùng không cần phải thao tác với chìa khóa hay ứng dụng để mở cửa.
* **Chức năng mã pin tạm thời:** Khi chúng ta vắng nhà và bận việc gì đó mà không thể mở khóa từ xa cho bạn bè người thân vào nhà nhưng không muốn cung cấp mã pin để tránh người đó vào nhà những lần sau mà không có sự cho phép , ta có thể tạo mã pin tạm thời và đặt thời gian hết hạn cho nó. Người khác vẫn có thể sử dụng mã pin để mở cửa nhưng chỉ trong 1 thời gian nhất định được xác định trước
* **Chức năng ghi lại hình ảnh lịch sử đóng mở cửa:** Nếu có hành động đóng mở cửa thì ESP32-CAM sẽ chụp ảnh lại và lưu lên trên cloud để người dùng có thể quan sát được lịch sử cũng như người mở cửa

# 

# Chương 2 Hệ thống phần cứng

## 2.1:Tổng quan các module được sử dụng trong dự án

**1. ESP32**

**Vai trò**: Bộ điều khiển chính.

**Mô tả**: ESP32 là một vi điều khiển tích hợp WiFi và Bluetooth, rất phù hợp cho các ứng dụng IoT. Nó có nhiều chân GPIO (General Purpose Input Output) để kết nối với các thiết bị ngoại vi như bàn phím, còi, màn hình LED, và công tắc giới hạn. ESP32 sẽ thực hiện các nhiệm vụ như đọc dữ liệu từ bàn phím, phát tín hiệu cho còi, hiển thị thông tin trên màn hình LED, và kiểm tra trạng thái của công tắc giới hạn. Đồng thời, nó sẽ giao tiếp với Firebase để lưu trữ và lấy dữ liệu mật khẩu, cũng như gửi lệnh điều khiển cửa.

**2. Bàn phím**

**Vai trò:** Thiết bị nhập mật khẩu.

**Mô tả:** Bàn phím ma trận (matrix keypad) thường được sử dụng trong các ứng dụng IoT để nhập dữ liệu. Nó bao gồm một lưới các nút bấm, mỗi nút kết nối với các chân GPIO của ESP32. Khi người dùng bấm các phím, ESP32 sẽ đọc tín hiệu và xác định phím nào được bấm. Mật khẩu được nhập từ bàn phím sẽ được so sánh với mật khẩu lưu trên Firebase.

**3. Còi (Buzzer)**

**Vai trò:** Thiết bị phát tín hiệu âm thanh.

**Mô tả:** Còi được sử dụng để cung cấp phản hồi âm thanh khi người dùng bấm bàn phím. Khi bất kỳ phím nào trên bàn phím được bấm, ESP32 sẽ gửi tín hiệu đến còi để phát ra âm thanh, báo hiệu rằng phím đã được nhận diện.

**4. Màn hình LED**

**Vai trò**: Thiết bị hiển thị thông tin.

**Mô tả**: Màn hình LED (loại 7-segment) sẽ hiển thị thông tin nhập từ bàn phím. ESP32 sẽ gửi dữ liệu đến màn hình LED để hiển thị từng ký tự khi người dùng bấm phím. Điều này giúp người dùng xác nhận rằng họ đã nhập đúng mật khẩu. Màn hình LED cũng có thể hiển thị thông báo trạng thái của cửa (mở/đóng).

**5. Công tắc giới hạn (Limit Switch)**

**Vai trò:** Kiểm tra trạng thái của cửa.

**Mô tả**: Công tắc giới hạn là một thiết bị cơ học có chức năng kiểm tra trạng thái đóng/mở của cửa. Khi cửa đóng hoặc mở, công tắc sẽ thay đổi trạng thái (ví dụ từ mở sang đóng). ESP32 sẽ đọc tín hiệu từ công tắc giới hạn để biết trạng thái hiện tại của cửa. Nếu công tắc từ mở chuyển sang đóng, ESP32 sẽ gửi lệnh lên Firebase để đóng cửa.

**6. Động cơ điều khiển ( Servo )**

**Mô tả**: Servo là một loại động cơ có khả năng điều khiển chính xác vị trí của trục quay. Nó thường được sử dụng trong các ứng dụng đòi hỏi độ chính xác cao như robot, mô hình điều khiển từ xa và hệ thống tự động.

**Vai trò**: Trong dự án này, servo sẽ thực hiện các lệnh điều khiển từ ESP32. Vị trí của servo sẽ được thay đổi dựa trên giá trị nhận được từ Firebase.

**Sơ đồ mạch tổng quan**

**A circuit board with wires

Description automatically generated**

*Hình 1. Sơ đồ module gửi lệnh điều khiển*

A circuit board with wires and a blue and red cable

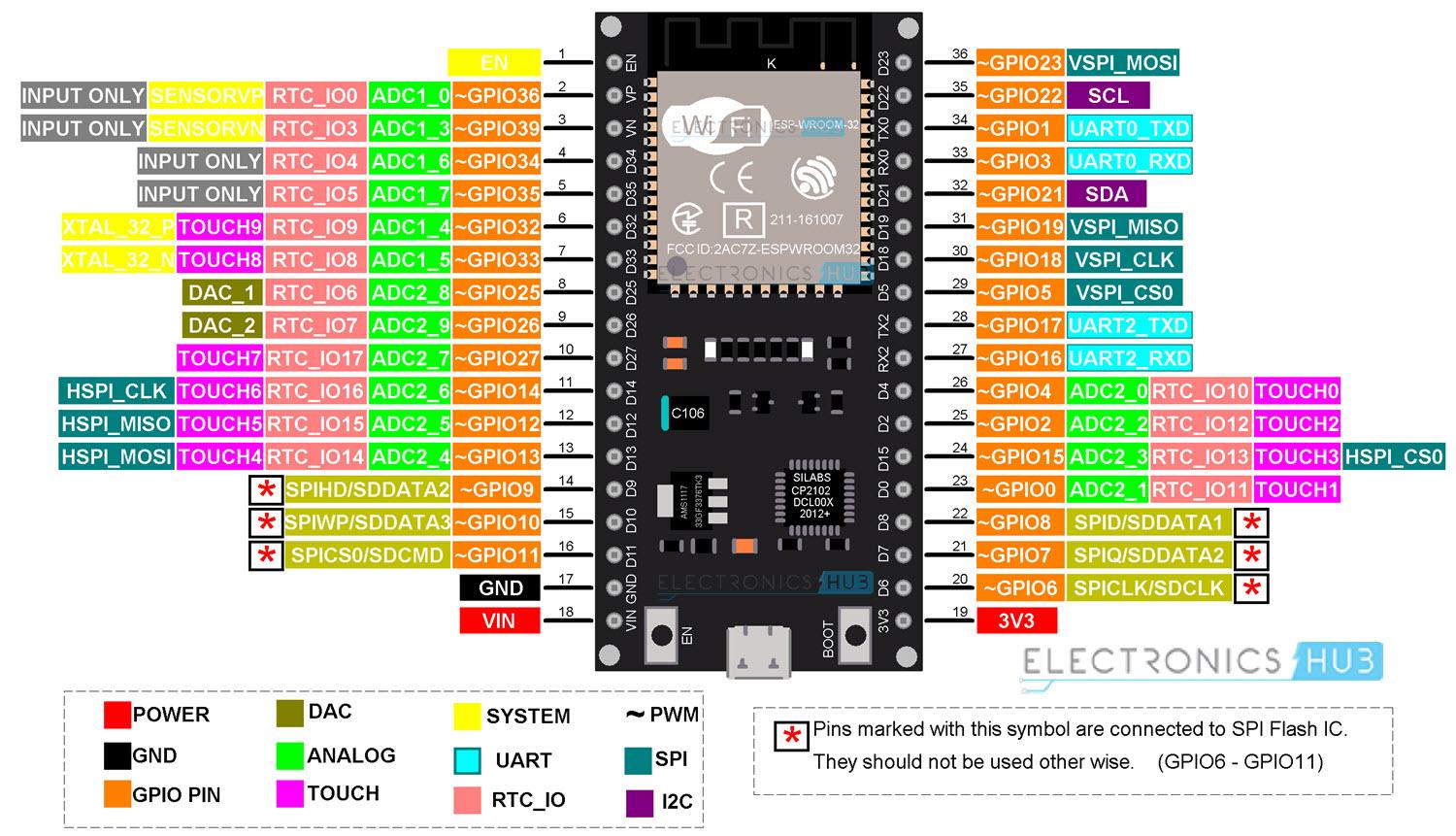
Description automatically generated with medium confidence

*Hình 2. Module thực hiện lệnh điều khiển*

## 2.2 Chi tiết về các module và chức năng

### 2.2.1.ESP32

**ESP32** là một module vi điều khiển tích hợp khả năng kết nối WiFi và Bluetooth, được phát triển bởi Espressif Systems. Đây là một trong những module được ưa chuộng nhất trong cộng đồng phát triển các ứng dụng Internet of Things (IoT) nhờ vào tính năng mạnh mẽ, tính linh hoạt và hiệu suất cao. Bài viết này sẽ cung cấp mô tả chi tiết về cấu trúc, tính năng, ứng dụng, cũng như cách lập trình và phát triển với ESP32.

**Sơ đồ chân ESP32**

Cấu trúc của ESP32

**1. Vi xử lý:**

* Lõi: ESP32 sử dụng lõi vi xử lý Xtensa LX6 32-bit với kiến trúc dual-core, có thể hoạt động độc lập hoặc cùng nhau để tăng cường hiệu suất.
* Tần số: Vi xử lý có thể đạt tốc độ tối đa lên đến 240 MHz, cung cấp khả năng xử lý mạnh mẽ cho các ứng dụng đòi hỏi.
* RAM: ESP32 được trang bị 520 KB SRAM, đủ để chạy các ứng dụng IoT phức tạp.
* Flash: Bộ nhớ Flash dao động từ 4MB đến 16MB, tùy thuộc vào phiên bản của module. Bộ nhớ này dùng để lưu trữ chương trình và dữ liệu.

**2. Kết nối:**

* WiFi: Hỗ trợ chuẩn IEEE 802.11 b/g/n, cho phép kết nối với mạng không dây. ESP32 có thể hoạt động ở cả hai chế độ Station và Access Point (AP), thậm chí hỗ trợ mạng WiFi Direct (P2P).
* Bluetooth: Hỗ trợ Bluetooth v4.2, bao gồm cả Bluetooth Classic (BR/EDR) và Bluetooth Low Energy (BLE). Điều này cho phép ESP32 giao tiếp với nhiều thiết bị Bluetooth khác nhau, từ các thiết bị di động đến các cảm biến BLE.

Tính năng nổi bật của ESP32

**1. Kết nối WiFi và Bluetooth:**

* ESP32 có khả năng kết nối mạnh mẽ và linh hoạt với cả WiFi và Bluetooth, cho phép dễ dàng truyền dữ liệu không dây và kết nối với nhiều thiết bị khác nhau.

**2. Hiệu suất cao:**

* Với vi xử lý dual-core 32-bit và tốc độ lên đến 240 MHz, ESP32 cung cấp hiệu suất xử lý mạnh mẽ, đáp ứng tốt các ứng dụng đòi hỏi hiệu năng cao như xử lý dữ liệu cảm biến, điều khiển thời gian thực, và truyền thông.

**3. Tiêu thụ điện năng thấp:**

* ESP32 được thiết kế với nhiều chế độ tiết kiệm năng lượng, từ chế độ Deep Sleep đến Light Sleep, giúp kéo dài thời gian hoạt động của các thiết bị sử dụng pin.

**4. Đa dạng giao tiếp ngoại vi:**

* ESP32 hỗ trợ nhiều giao tiếp như SPI, I2C, UART, I2S, CAN, và PWM, cho phép kết nối và điều khiển nhiều loại thiết bị ngoại vi khác nhau.

Ứng dụng của ESP32

**1. Thiết bị nhà thông minh:**

* Điều khiển đèn, quạt, ổ cắm điện thông minh và các thiết bị gia dụng khác thông qua WiFi hoặc Bluetooth.
* Hệ thống giám sát và điều khiển từ xa các thiết bị gia dụng thông qua ứng dụng di động hoặc trình duyệt web.

**2. Hệ thống an ninh:**

* Camera IP, hệ thống cảnh báo, cảm biến chuyển động và các thiết bị an ninh khác.
* Hệ thống giám sát và cảnh báo qua mạng WiFi hoặc Bluetooth, gửi thông báo đến điện thoại hoặc máy tính khi có sự cố.

**3. Thiết bị đeo:**

* Vòng tay sức khỏe, đồng hồ thông minh, thiết bị theo dõi thể thao, và các thiết bị đeo khác.
* Thu thập và truyền dữ liệu sức khỏe, hoạt động thể dục, và các thông tin cá nhân khác qua Bluetooth.

**4. Ứng dụng công nghiệp:**

* Cảm biến, hệ thống điều khiển từ xa, tự động hóa công nghiệp, và các ứng dụng IoT công nghiệp khác.
* Kết nối và điều khiển các thiết bị công nghiệp qua mạng WiFi, giám sát và quản lý dữ liệu từ xa.

Lập trình và phát triển với ESP32

**1.Arduino IDE:**

* Phổ biến và dễ sử dụng: Arduino IDE là môi trường lập trình phổ biến và dễ sử dụng, đặc biệt là cho người mới bắt đầu.
* Thư viện phong phú: Hỗ trợ nhiều thư viện cho WiFi, Bluetooth, GPIO, và các giao tiếp khác, giúp đơn giản hóa việc lập trình các tính năng của ESP32.

**2. ESP-IDF (Espressif IoT Development Framework):**

* Môi trường chính thức: ESP-IDF là môi trường phát triển chính thức từ Espressif, cung cấp đầy đủ các tính năng và tài liệu hướng dẫn chi tiết.
* Mạnh mẽ và linh hoạt**:** ESP-IDF cho phép cấu hình chi tiết và tối ưu hóa cho các dự án lớn, hỗ trợ các tính năng nâng cao như OTA (Over-the-Air) updates, bảo mật, và quản lý kết nối.

**3. MicroPython:**

* Lập trình bằng Python: MicroPython cho phép lập trình ESP32 bằng ngôn ngữ Python, dễ học và sử dụng cho các dự án đơn giản và nhanh chóng.
* Thư viện hỗ trợ: MicroPython cung cấp các thư viện cho WiFi, GPIO, và các giao tiếp khác, giúp lập trình dễ dàng hơn.

**4. PlatformIO:**

* IDE mạnh mẽ: PlatformIO là một môi trường phát triển tích hợp (IDE) mạnh mẽ, hỗ trợ nhiều nền tảng và framework, tích hợp tốt với các công cụ phát triển khác như Visual Studio Code.
* Dễ quản lý dự án: PlatformIO hỗ trợ quản lý thư viện, xây dựng và triển khai dự án một cách hiệu quả, giúp tiết kiệm thời gian và công sức trong quá trình phát triển.

Cách sử dụng ESP32

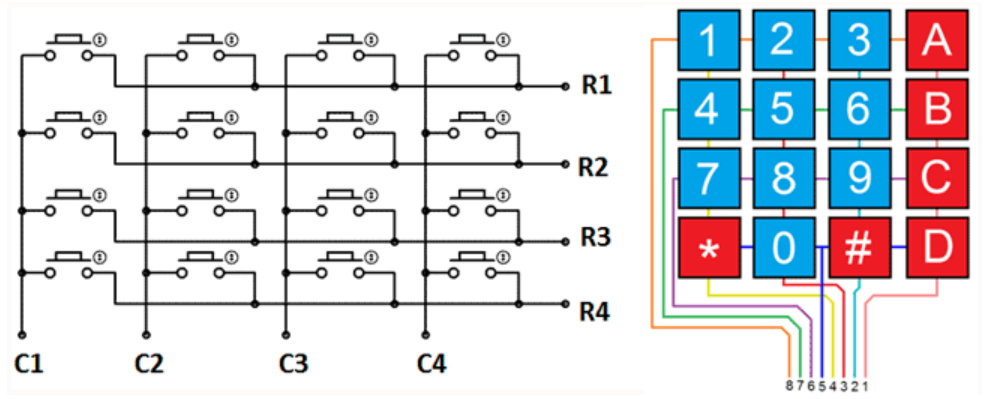
**1. Kết nối và cấu hình ban đầu:**

* Kết nối với máy tính: Sử dụng cáp USB để kết nối ESP32 với máy tính. Đảm bảo cài đặt đúng driver (nếu cần) để máy tính nhận diện ESP32.
* Cài đặt môi trường phát triển: Tải và cài đặt Arduino IDE, ESP-IDF, hoặc PlatformIO tùy theo nhu cầu và sở thích của bạn.
* Chọn board và cổng COM**:** Trong môi trường phát triển, chọn đúng board (ESP32) và cổng COM mà ESP32 đang kết nối.

**2. Viết và tải chương trình:**

* Viết mã nguồn: Sử dụng các thư viện và hàm có sẵn để lập trình các tính năng cần thiết. Ví dụ, trong Arduino IDE, bạn có thể sử dụng thư viện WiFi.h để kết nối WiFi, hay thư viện BluetoothSerial.h để sử dụng Bluetooth.
* Tải chương trình lên ESP32: Sử dụng công cụ tải (upload) trong môi trường phát triển để nạp chương trình vào ESP32. Đảm bảo rằng chương trình không có lỗi trước khi tải lên.
* Kiểm tra và debug: Sử dụng các công cụ debug như Serial Monitor trong Arduino IDE để kiểm tra và sửa lỗi chương trình. Bạn có thể gửi và nhận dữ liệu qua cổng nối tiếp để kiểm tra hoạt động của ESP32.

### 2.2.2.Bàn phím



**Vai trò:** Thiết bị nhập mật khẩu.

**Mô tả:** Module bàn phím, thường là bàn phím ma trận (matrix keypad), là một thiết bị phổ biến trong các dự án điện tử và IoT để nhập dữ liệu. Nó bao gồm một lưới các nút bấm được sắp xếp theo hàng và cột, tạo thành một ma trận. Khi người dùng bấm phím, bàn phím sẽ gửi tín hiệu đến vi điều khiển (trong trường hợp này là ESP32), cho phép vi điều khiển biết được phím nào đã được bấm.

**Cấu trúc của Bàn phím Ma trận**

Một bàn phím ma trận 4x4 bao gồm 4 hàng và 4 cột, tổng cộng có 16 nút bấm. Mỗi nút bấm nằm tại giao điểm của một hàng và một cột.

* Hàng (Rows): Các chân kết nối với các hàng của bàn phím.
* Cột (Columns): Các chân kết nối với các cột của bàn phím.

**Cách hoạt động**

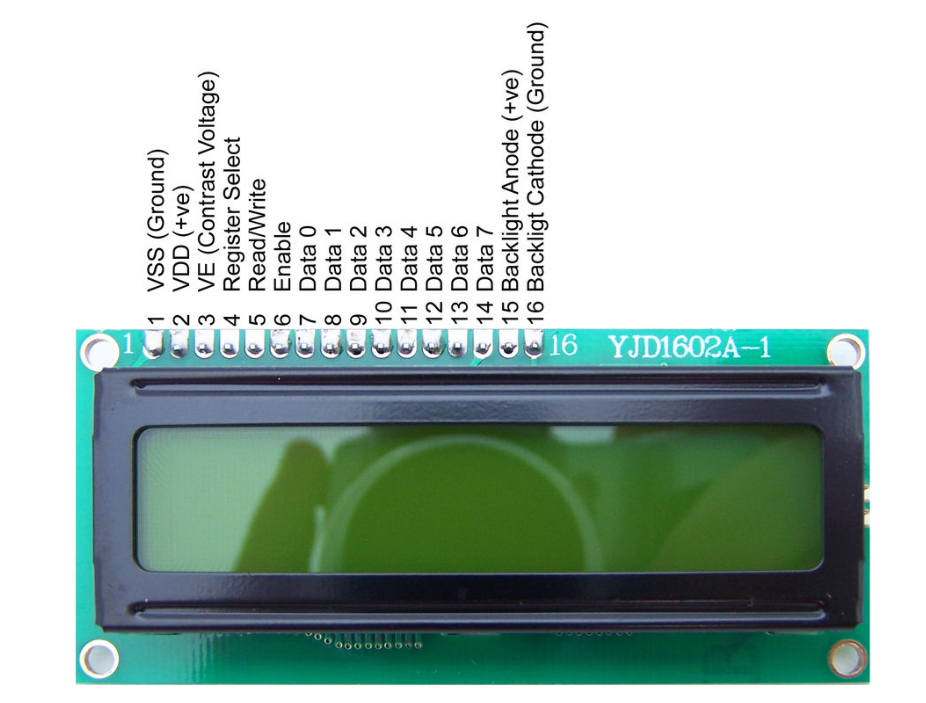
Khi một phím được bấm, nó sẽ kết nối một hàng với một cột. Vi điều khiển sẽ lần lượt kiểm tra các hàng và cột để xác định phím nào đã được bấm. Điều này được thực hiện thông qua quá trình quét (scanning):

1. Quét hàng: Vi điều khiển kích hoạt lần lượt từng hàng một.
2. Kiểm tra cột: Tại mỗi hàng, vi điều khiển sẽ kiểm tra từng cột để xem có tín hiệu hay không.

Ví dụ, nếu phím '5' được bấm trên bàn phím 4x4:

* Hàng 2 sẽ được kích hoạt.
* Tín hiệu sẽ được phát hiện tại cột 2.
* Vi điều khiển sẽ xác định rằng phím '5' tại hàng 2 và cột 2 đã được bấm.

### 2.2.3.Màn hình Led



**Màn Hình LED OLED**

**Vai trò:** Hiển thị thông tin văn bản và đồ họa chi tiết.

**Mô tả:** Màn hình OLED (Organic Light Emitting Diode) là một loại màn hình sử dụng công nghệ đi-ốt phát quang hữu cơ để hiển thị hình ảnh. Màn hình OLED có nhiều ưu điểm so với các loại màn hình truyền thống như LCD, bao gồm độ sáng cao, góc nhìn rộng, độ tương phản cao và khả năng hiển thị màu sắc sống động.

**Cấu trúc của Màn Hình OLED**

1. Cấu tạo pixel: Mỗi pixel của màn hình OLED bao gồm một lớp vật liệu hữu cơ phát quang được kẹp giữa hai điện cực. Khi dòng điện chạy qua, vật liệu hữu cơ phát sáng, tạo ra hình ảnh trực tiếp mà không cần đèn nền.
2. Điều khiển**:** Màn hình OLED thường sử dụng các bộ điều khiển giao tiếp qua giao thức I2C hoặc SPI, giúp giảm số lượng chân kết nối cần thiết và đơn giản hóa quá trình lập trình.

**Kết Nối Màn Hình OLED với ESP32**

**Kết nối I2C**

* SCL: Chân Clock (thường là GPIO 22 trên ESP32).
* SDA: Chân Data (thường là GPIO 21 trên ESP32).
* VCC: Nguồn (3.3V hoặc 5V tùy màn hình).
* GND: Mass.

**Các Ứng Dụng**

* Hiển thị dữ liệu thời gian thực: Giá trị cảm biến, thông báo trạng thái hệ thống.
* Giao diện người dùng: Hướng dẫn, thông báo, và phản hồi từ hệ thống.
* Trình bày đồ họa: Biểu đồ, hình ảnh đơn giản, biểu tượng.

**Lưu Ý**

* Nguồn điện: Đảm bảo cung cấp đủ nguồn điện cho màn hình OLED. Một số màn hình có thể yêu cầu 3.3V hoặc 5V.
* Giao tiếp đúng cách: Sử dụng giao thức phù hợp (I2C hoặc SPI) và kết nối chính xác.
* Thư viện hỗ trợ: Sử dụng các thư viện đã được kiểm chứng như Adafruit\_GFX và Adafruit\_SSD1306 để đơn giản hóa việc lập trình và điều khiển màn hình.

**Ưu Điểm**

* Chất lượng hiển thị cao: Độ tương phản và độ sáng tốt, hiển thị màu sắc chính xác.
* Tiết kiệm năng lượng: Mỗi pixel tự phát sáng nên chỉ tiêu thụ năng lượng khi hiển thị.
* Mỏng và nhẹ: Thiết kế mỏng, nhẹ, phù hợp cho các thiết bị di động.

**Nhược Điểm**

* Tuổi thọ hạn chế: Các vật liệu hữu cơ có thể bị lão hóa theo thời gian, giảm độ sáng.
* Giá thành cao: So với màn hình LCD, màn hình OLED thường có giá cao hơn.

### 2.2.4.Còi



**Vai trò:** Module còi được sử dụng để tạo ra âm thanh cảnh báo hoặc phản hồi trong các hệ thống IoT. Trong dự án, còi sẽ kêu khi bấm bàn phím để báo hiệu cho người dùng rằng thao tác đã được nhận.

**Các Loại Module Còi**

1. **Còi Thụ Động (Passive Buzzer):**
   * Nguyên lý hoạt động: Còi thụ động cần một tín hiệu PWM (Pulse Width Modulation) để tạo ra âm thanh. Điều này có nghĩa là bạn cần phải cung cấp tần số dao động để còi phát ra âm thanh.
   * Ưu điểm: Có thể tạo ra nhiều loại âm thanh khác nhau bằng cách thay đổi tần số.
   * Nhược điểm: Yêu cầu điều khiển phức tạp hơn.
2. **Còi Chủ Động (Active Buzzer):**
   * Nguyên lý hoạt động: Còi chủ động có mạch dao động tích hợp bên trong, chỉ cần cung cấp nguồn điện hoặc tín hiệu logic (HIGH/LOW) để phát ra âm thanh cố định.
   * Ưu điểm: Dễ dàng sử dụng, chỉ cần cấp nguồn hoặc tín hiệu logic.
   * Nhược điểm: Chỉ phát ra một loại âm thanh cố định.

**Kết Nối Module Còi với ESP32**

**Kết Nối Còi Chủ Động**

* Chân VCC: Kết nối với nguồn (3.3V hoặc 5V tùy còi).
* Chân GND: Kết nối với mass.
* Chân Signal: Kết nối với chân GPIO trên ESP32 để điều khiển.

**Kết Nối Còi Thụ Động**

* Chân VCC: Kết nối với nguồn (3.3V hoặc 5V tùy còi).
* Chân GND: Kết nối với mass.
* Chân Signal: Kết nối với chân PWM trên ESP32 để tạo tín hiệu điều khiển.

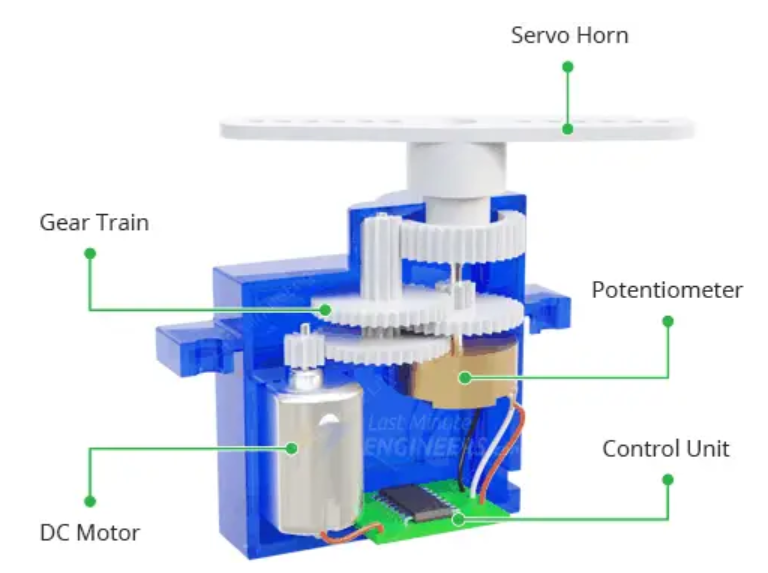
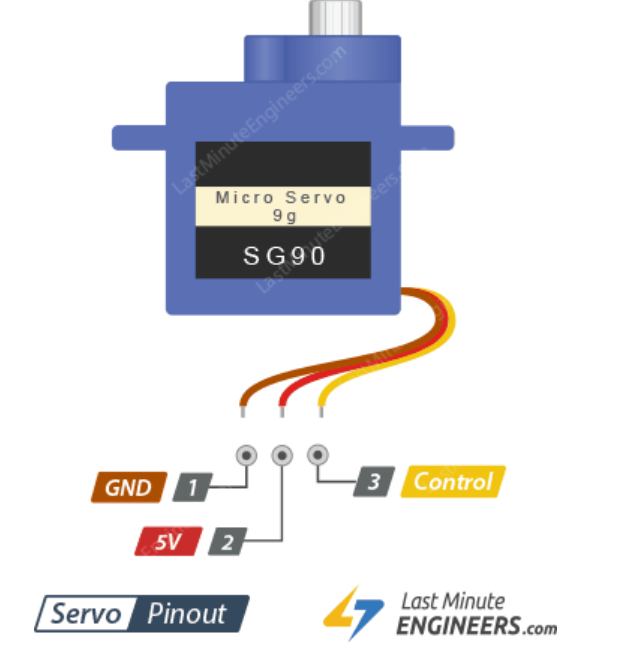
**Các Ứng Dụng Của Còi Trong Dự Án IoT**

1. Cảnh báo âm thanh: Sử dụng còi để cảnh báo người dùng về các sự kiện quan trọng như báo động an ninh, vượt ngưỡng cảm biến, hoặc lỗi hệ thống.
2. Phản hồi thao tác: Phát ra âm thanh để phản hồi các thao tác của người dùng như bấm nút hoặc nhập dữ liệu.
3. Âm thanh thông báo: Sử dụng còi để thông báo trạng thái hoạt động của hệ thống, ví dụ như khởi động hoặc tắt máy.

**Lưu Ý Khi Sử Dụng Còi**

1. Nguồn điện: Đảm bảo rằng còi được cấp nguồn đúng yêu cầu để tránh hư hỏng.
2. Âm lượng: Một số còi có thể phát ra âm thanh rất lớn, cần đặt ở vị trí phù hợp để không gây phiền hà.
3. Tần số điều khiển (đối với còi thụ động): Chọn tần số điều khiển phù hợp để tạo ra âm thanh mong muốn.

### 2.2.5. Servo



**Vai trò:** Module servo được sử dụng để điều khiển các chuyển động chính xác trong các hệ thống IoT. Trong dự án của bạn, module servo có thể được sử dụng để điều khiển các cơ cấu như cửa (mở/đóng) theo lệnh từ hệ thống.

**Khái Niệm Về Servo**

Servo là một thiết bị điện cơ, có khả năng di chuyển theo góc xác định (thường trong khoảng từ 0° đến 180° hoặc 360° tùy loại servo). Điều khiển servo thường thông qua tín hiệu PWM (Pulse Width Modulation), với độ dài xung PWM quyết định góc quay của servo.

**Các Loại Servo**

1. **Servo Phản Hồi Vị Trí (Position Control Servo):**
   * Loại phổ biến nhất, điều khiển góc quay của servo bằng tín hiệu PWM. Góc quay được điều khiển trong một phạm vi nhất định (thường từ 0° đến 180° hoặc 0° đến 360°).
2. **Servo Lực (Torque Servo):**
   * Loại servo có khả năng tạo ra lực quay lớn, thường dùng trong các ứng dụng công nghiệp hoặc robot.
3. **Servo Tốc Độ (Speed Servo):**
   * Điều khiển tốc độ quay của servo thay vì góc quay. Điều này cho phép kiểm soát tốc độ di chuyển của các bộ phận cơ khí.

**Các Thành Phần Của Servo**

* Motor: Là thành phần chính tạo ra chuyển động.
* Potentiometer (Biến trở): Được sử dụng để đo góc quay của motor.
* Mạch điều khiển: Điều chỉnh động cơ sao cho đạt được góc quay chính xác.

**Kết Nối Servo Với ESP32**

Servo được kết nối với ESP32 qua một chân GPIO sử dụng tín hiệu PWM để điều khiển góc quay. Cần lưu ý nguồn điện cung cấp cho servo, vì một số loại servo cần nguồn điện cao hơn nguồn 3.3V của ESP32.

**Kết Nối**

1. VCC (Nguồn): Kết nối với nguồn (thường là 5V hoặc 6V tùy loại servo).
2. GND (Mass): Kết nối với mass của ESP32 và nguồn.
3. Signal (Tín hiệu): Kết nối với một chân GPIO trên ESP32 (ví dụ GPIO 15).

**Các Lệnh Cơ Bản Điều Khiển Servo**

* myServo.attach(pin);: Kết nối đối tượng servo với chân GPIO. pin là chân GPIO mà bạn kết nối với tín hiệu của servo.
* myServo.write(angle);: Điều khiển servo quay đến góc angle. Góc có thể là từ 0° đến 180°.
* myServo.writeMicroseconds(microseconds);: Thay vì sử dụng góc, bạn có thể điều khiển servo bằng cách sử dụng thời gian của xung PWM (thời gian này được tính bằng micro giây).
* myServo.read();: Đọc giá trị góc hiện tại mà servo đang quay.
* myServo.detach();: Ngừng điều khiển servo.

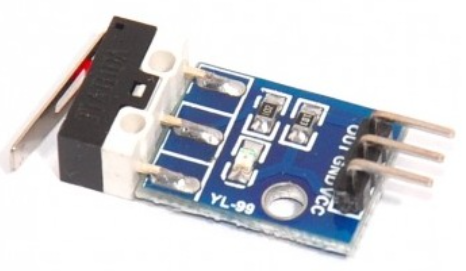
**Các Ứng Dụng của Servo**

1. **Điều Khiển Cửa (Locking System):**
   * Trong hệ thống IoT của bạn, servo có thể được sử dụng để mở và đóng cửa bằng cách điều khiển góc quay của servo (có thể là 0° cho đóng và 90° cho mở).
2. **Robot:**
   * Servo được sử dụng trong các dự án robot để điều khiển các khớp và bộ phận chuyển động của robot.
3. **Mô Phỏng Chuyển Động Cơ Học:**
   * Servo có thể được sử dụng để mô phỏng các chuyển động trong các dự án như mô phỏng cơ cấu máy hoặc mô hình điều khiển tự động.

**Lưu Ý Khi Sử Dụng Servo**

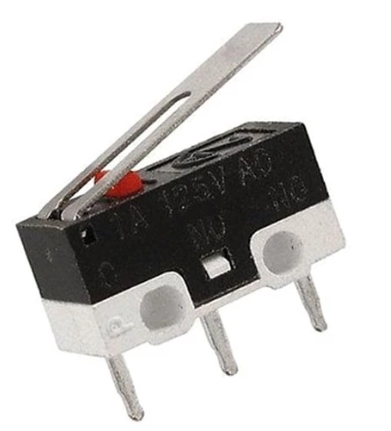
1. **Nguồn Cung Cấp:**
   * Servo thường cần một nguồn điện từ 4.5V đến 6V, tùy thuộc vào loại servo. Đảm bảo cấp nguồn đúng cho servo vì nguồn từ ESP32 (3.3V) có thể không đủ mạnh để điều khiển servo.
2. **Dòng Điện:**
   * Servo có thể tiêu thụ dòng điện lớn khi hoạt động, đặc biệt là khi tải của servo nặng. Đảm bảo nguồn cung cấp có khả năng cung cấp đủ dòng điện.
3. **Tốc Độ và Góc Quay:**
   * Các servo có tốc độ quay và độ chính xác góc khác nhau. Tốc độ có thể ảnh hưởng đến các ứng dụng cần độ chính xác cao hoặc chuyển động nhanh.
4. **Khả Năng Vượt Quá Góc Quay:**
   * Đảm bảo servo không bị buộc phải quay quá góc quy định (thường là 180°). Điều này có thể gây hư hỏng cho servo.

### 2.2.6. Công tắc giới hạn (Limit Switch)



Công tắc giới hạn (limit switch) là một thiết bị điện được sử dụng để phát hiện vị trí của một đối tượng di chuyển, thường được ứng dụng trong các hệ thống tự động hóa, máy móc, hoặc các dự án DIY. Công tắc giới hạn với 3 chân (1 chân VCC, 1 chân GND, và 1 chân DO - Digital Output) rất phổ biến trong các ứng dụng IoT và điều khiển tự động.

**Sơ đồ chân**

****

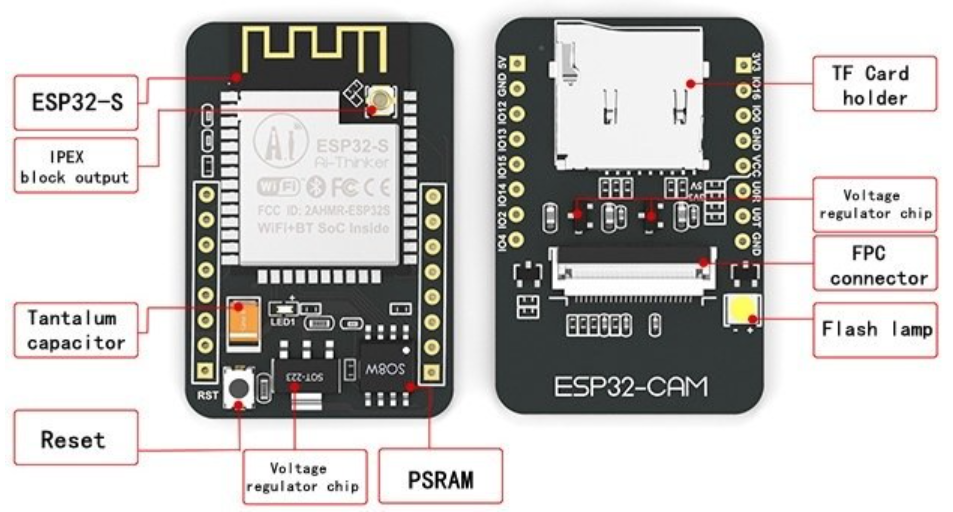
**Mô tả tổng quan**

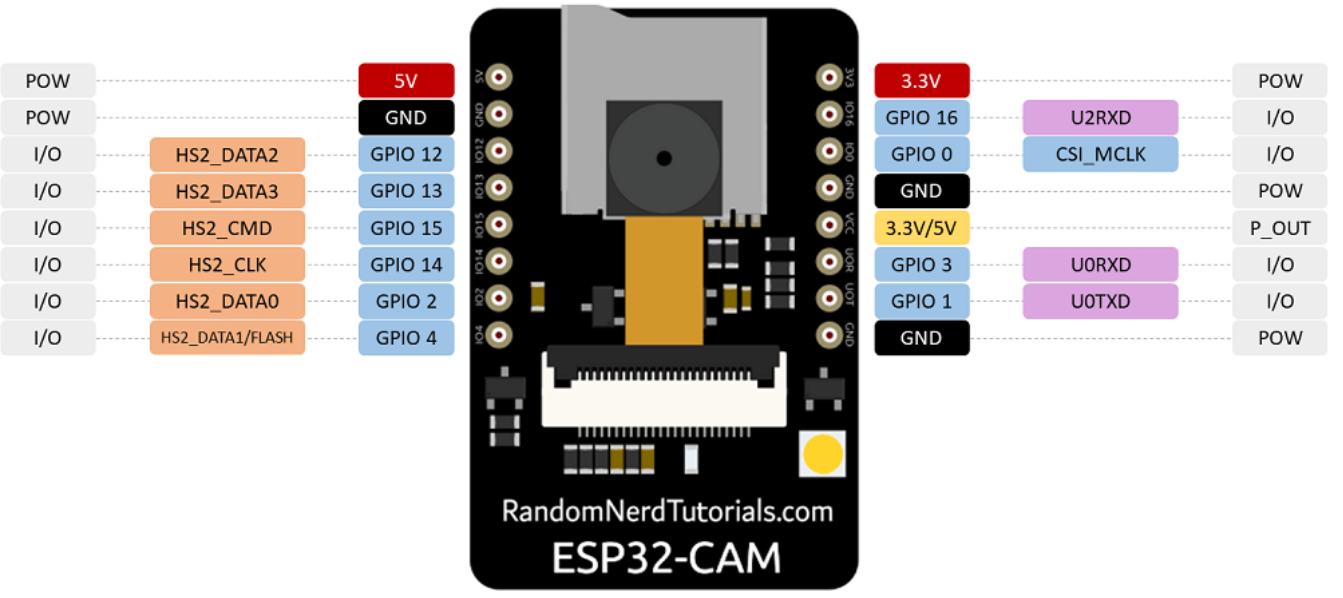
* **Chức năng**: Công tắc giới hạn hoạt động như một công tắc điện tử có khả năng phát hiện sự hiện diện hoặc vị trí của một đối tượng. Khi đối tượng (như một cửa hoặc trục) chạm vào công tắc, nó sẽ thay đổi trạng thái từ "mở" sang "đóng" (hoặc ngược lại) và gửi tín hiệu đến vi điều khiển hoặc mạch điều khiển.
* **Ứng dụng**: Công tắc giới hạn thường được sử dụng trong các ứng dụng như cảm biến vị trí trong robot, thiết bị điều khiển cửa tự động, máy in 3D, và các hệ thống bảo vệ trong công nghiệp.

**Nguyên lý hoạt động**

* Khi công tắc giới hạn chưa được kích hoạt, chân DO thường ở trạng thái LOW. Khi một đối tượng chạm vào công tắc, công tắc sẽ đóng mạch và chân DO sẽ chuyển sang trạng thái HIGH. Ngược lại, khi đối tượng rời khỏi công tắc, chân DO sẽ quay trở lại trạng thái LOW.
* Tín hiệu từ chân DO có thể được đọc bởi vi điều khiển để xác định vị trí của đối tượng và thực hiện các hành động tương ứng.

### 2.2.7. ESP32 CAM





**ESP32-CAM** là một module phát triển nhỏ gọn và mạnh mẽ, sử dụng vi điều khiển ESP32 với khả năng kết nối Wi-Fi và Bluetooth, đi kèm với một camera tích hợp (OV2640) và nhiều chân GPIO để kết nối với các thiết bị ngoại vi khác. Đây là một module rất phổ biến trong các ứng dụng IoT (Internet of Things), đặc biệt là trong các dự án yêu cầu chức năng ghi hình, nhận dạng hình ảnh hoặc giám sát từ xa.

**Các Thành Phần Chính**

1. **ESP32 Chipset:**
   * ESP32 là một vi điều khiển hai nhân với khả năng xử lý mạnh mẽ, tích hợp Wi-Fi và Bluetooth. Đây là nền tảng lý tưởng cho các ứng dụng IoT, nhờ vào tốc độ xử lý cao và khả năng kết nối không dây.
2. **Camera OV2640:**
   * Đây là một camera 2MP với độ phân giải tối đa 1600x1200 pixel. OV2640 rất phổ biến trong các ứng dụng camera nhúng nhờ vào giá thành rẻ và khả năng xử lý hình ảnh khá tốt.
3. **Chân GPIO:**
   * ESP32-CAM có nhiều chân GPIO (General Purpose Input/Output), cho phép kết nối với các thiết bị ngoại vi như cảm biến, công tắc, động cơ, LED và nhiều thiết bị khác.
4. **Thẻ nhớ MicroSD:**
   * ESP32-CAM hỗ trợ thẻ nhớ MicroSD, cho phép lưu trữ dữ liệu hình ảnh hoặc video trực tiếp vào thẻ nhớ. Điều này đặc biệt hữu ích trong các ứng dụng giám sát, nơi bạn muốn lưu lại video mà không cần kết nối trực tiếp với máy tính hoặc thiết bị lưu trữ khác.
5. **Chân kết nối JTAG:**
   * ESP32-CAM có các chân JTAG cho phép gỡ lỗi phần mềm và kiểm tra chương trình trong quá trình phát triển.
6. **Chân nguồn (5V và 3.3V):**
   * Cung cấp điện cho ESP32-CAM, thường là nguồn 5V cho board và 3.3V cho các thành phần khác.

**Các Ứng Dụng Của ESP32-CAM**

1. **Giám Sát Video (CCTV):**
   * ESP32-CAM có thể được sử dụng trong các hệ thống giám sát video, nơi nó có thể ghi hình và truyền tải video qua Wi-Fi đến các thiết bị khác như điện thoại di động hoặc máy tính.
2. **Nhận Dạng Khuôn Mặt (Face Recognition):**
   * ESP32-CAM có thể được sử dụng trong các hệ thống nhận dạng khuôn mặt, kết hợp với phần mềm như OpenCV để nhận diện người trong ảnh hoặc video. Điều này có thể ứng dụng trong các hệ thống an ninh, kiểm soát ra vào, hoặc xác thực người dùng.
3. **Camera Web (Webcam):**
   * ESP32-CAM có thể được sử dụng như một webcam để truyền hình ảnh/video trực tiếp lên các nền tảng như YouTube hoặc các dịch vụ video streaming khác.
4. **Hệ Thống Điều Khiển Từ Xa (Remote Control):**
   * ESP32-CAM có thể tích hợp với các ứng dụng di động hoặc hệ thống máy tính để điều khiển từ xa thiết bị hoặc nhận thông tin video trực tiếp từ xa.
5. **Chế Tạo Robot Thị Giác:**
   * Với khả năng ghi hình và truyền tải dữ liệu hình ảnh, ESP32-CAM có thể được sử dụng trong các robot thị giác máy tính để nhận diện môi trường và đưa ra quyết định.

**Cách Kết Nối ESP32-CAM**

ESP32-CAM yêu cầu một số bước cụ thể để kết nối và lập trình, vì nó không có cổng USB trực tiếp, và cần phải sử dụng bộ chuyển đổi USB-to-Serial (USB-TTL).

**Các bước kết nối cơ bản:**

1. **Chân Kết Nối:**
   * **GND (Ground):** Kết nối với GND của bộ chuyển đổi USB-to-Serial.
   * **5V (Nguồn):** Kết nối với nguồn 5V của bộ chuyển đổi USB-to-Serial.
   * **U0R (RX của ESP32-CAM):** Kết nối với TX của bộ chuyển đổi USB-to-Serial.
   * **U0T (TX của ESP32-CAM):** Kết nối với RX của bộ chuyển đổi USB-to-Serial.
2. **Chế Độ Nạp Lập Trình:**
   * Khi nạp chương trình vào ESP32-CAM, bạn cần phải nhấn nút **"Boot"** trên module ESP32-CAM để đưa nó vào chế độ nạp chương trình.
3. **Cài Đặt Arduino IDE:**
   * Để lập trình ESP32-CAM, bạn cần cài đặt **Arduino IDE** và thêm ESP32 vào danh sách board của IDE.
   * Cài đặt thư viện hỗ trợ camera (thư viện esp32-cam).

**Chức năng trong dự án :**

* ESP32-CAM sẽ chụp ảnh lại nếu có ai thực hiện đóng mở cửa để lưu lại trên cloud để người dùng có thể xem lại ai đã mở cửa để đảm bảo an ninh , an toàn nếu có người lạ vào.

## 2.3 Mô tả về cách hoạt động và mã nguồn Arduino

### 2.3.1. Sơ đồ mạch tổng quan

A circuit board with wires

Description automatically generated

A circuit board with wires and a blue and red cable

Description automatically generated with medium confidence

### 2.3.2.Mô tả nguyên tắc hoạt động

**Nguyên tắc hoạt động:**

**- Module gửi lệnh điều khiển:**

+ Khóa cửa : Module sẽ nhận tín hiệu từ công tắc giới hạn , nếu đóng cửa ( công tắc giới hạn mà đóng ) sẽ có cơ chế tự khóa cửa khi đóng , module sẽ tự động gửi tín hiệu khóa cửa lên firebase.

+ Mở cửa : Module sẽ nhận tín hiệu từ bàn phím , nếu bấm phím A chúng ta sẽ bắt đầu quá trình nhập bàn phím, thông tin nhập sẽ được hiển thị trên màn hình , độ dài của mật khẩu là 4 , nếu đủ số lượng ký tự thì sẽ bắt đầu quá trình lấy dữ liệu password trên firebase , sau đó thực hiện so sánh nếu đúng sẽ gửi lệnh mở cửa lên firebase

**- Module thực hiện lệnh điều khiển:**

**+** Esp32 sẽ liên tục lấy thông tin từ phía firebase , nếu có sự thay đổi về trạng thái thì module sẽ thực hiện các lệnh điều khiển tương ứng , nếu khóa thì quay servo 1 góc 180 độ , ngược lại nếu mở thì quay servo với một góc 90 độ

+ Với cách này chúng ta hoàn toàn có thể điều khiển cửa từ xa thông qua mạng internet bằng ứng dụng web/app mà chúng em xây dựng hoặc có thể sử dụng trực tiếp

* **Module ESP32-CAM :**
* **ESP32-CAM** sẽ tạo ra một endpoint được publish qua mạng internet , nếu có hành động đóng mở rèm thì module thực hiện lệnh điều khiển sẽ gọi đến endpoint này lấy ảnh để cập nhật lên cloud

### 2.3.3. Mã nguồn Aduino

**Mã nguồn module gửi lệnh điều khiển**

| #include <Keypad.h>  #include <ESP32Servo.h>  #include <Arduino.h>  #include <LiquidCrystal\_I2C.h>  #include <WiFi.h>  #include <HTTPClient.h>  #include <ArduinoJson.h>  // #include <NTPClient.h>  // #include <WiFiUdp.h>  // #include <TimeLib.h>  // wifi  const char\* ssid = "Duong";  const char\* passwordwf = "11111111";  // Thông tin Firebase  const char\* FIREBASE\_HOST = "https://bt-iot-7af8a-default-rtdb.asia-southeast1.firebasedatabase.app/";  const char\* FIREBASE\_AUTH = "i0SPueav25FRwD0lKD7BiQjGE9TVL9lVyhcuoDid";  const byte ROWS = 4; // số hàng của bàn phím  const byte COLS = 4; // số cột của bàn phím  #define LIMIT\_SWITCH\_PIN 23  int lcdColumns = 16;  int lcdRows = 2;  LiquidCrystal\_I2C lcd(0x27, lcdColumns, lcdRows);  // Định nghĩa các chân của bàn phím  char keys[ROWS][COLS] = {  { '1', '2', '3', 'A' },  { '4', '5', '6', 'B' },  { '7', '8', '9', 'C' },  { '\*', '0', '#', 'D' }  };  byte rowPins[ROWS] = { 14, 27, 26, 25 }; // Sử dụng các chân GPIO khác  byte colPins[COLS] = { 33, 32, 18, 19 }; // Sử dụng các chân GPIO khác  Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);  Servo myServo;  const int servoPin = 4; // chân kết nối servo  const int buzzerPin = 4; // chân kết nối buzzer  // const int switchPin = 34;  // String password = "1245"; // mật khẩu đúng  String inputPassword = "";  bool isEnteringPassword = false;  int prevSwitchState = -1;  int count = 0;  // WiFiUDP udp;  // Khởi tạo đối tượng NTPClient  // NTPClient timeClient(udp, "pool.ntp.org", 25200);  void setup() {  Serial.begin(115200);  // Bắt đầu NTPClient  // timeClient.begin();  myServo.attach(servoPin);  // Kết nối WiFi  WiFi.begin(ssid, passwordwf);  while (WiFi.status() != WL\_CONNECTED) {  delay(1000);  Serial.println("Connecting to WiFi...");  }  Serial.println("Connected to WiFi");  pinMode(buzzerPin, OUTPUT);  pinMode(LIMIT\_SWITCH\_PIN, INPUT\_PULLDOWN);  // Khởi tạo LCD  lcd.init();  // Bật đèn nền LCD  lcd.backlight();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap");  }  void loop() {  // handleTime();  int switchState = digitalRead(LIMIT\_SWITCH\_PIN);  if (prevSwitchState == -1) {  prevSwitchState = switchState;  }  Serial.println(switchState);  if (switchState == 1) {  if (prevSwitchState != switchState) {  prevSwitchState = switchState;  // delay(500);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Khoa cua ...");  updateLEDStatus("OFF");  tone(buzzerPin, 3000, 300);  tone(buzzerPin, 3000, 600);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap:");  }  }  char key = keypad.getKey();  if (key) {  tone(buzzerPin, 3000, 100); // bật buzzer mỗi lần nhấn phím  Serial.println(key);  if (key == 'A') {  if (switchState == 0) {  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Cua dang mo");  delay(500);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap:");  return;  }  Serial.println(switchState);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhap mat khau:");  isEnteringPassword = true;  inputPassword = "";  Serial.println("Start entering password");  } else if (key == 'C') {  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Huy....");  delay(500);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap");  isEnteringPassword = false;  inputPassword = "";  Serial.println("Cancelled");  } else if (isEnteringPassword) {  inputPassword += key;  // lcd.clear();  lcd.setCursor(count, 1);  lcd.print(key);  count++;  // Kiểm tra khi người dùng nhập đủ độ dài mật khẩu  if (inputPassword.length() == 4) {  // handleTime();  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Loading ...");  String password = getPassword();  String passwordTmp = getPasswordTmp();  passwordTmp = passwordTmp.substring(1, 5);  password = password.substring(1, 5);  Serial.print("Password cloud: ");  Serial.println(password);  Serial.print("Password cloud tmp: ");  Serial.println(passwordTmp);  Serial.print("Password: ");  Serial.println(inputPassword);  if (inputPassword == password || inputPassword == passwordTmp) {  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Thanh Cong...");  prevSwitchState = 0;  updateLEDStatus("ON");  delay(5000);  switchState = digitalRead(LIMIT\_SWITCH\_PIN);  if (switchState == 1) {  Serial.print("state");  Serial.print(switchState);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Khoa cua ...");  updateLEDStatus("OFF");  tone(buzzerPin, 3000, 300);  tone(buzzerPin, 3000, 600);  }  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap");  Serial.println("Access Granted");  tone(4, 3000, 1000);  Serial.println("Access Granted");  } else {  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("That Bai");  delay(500);  lcd.clear();  lcd.setCursor(0, 0);  lcd.print("Nhan A de nhap");  Serial.println("Access Denied");  }  count = 0;  inputPassword = ""; // reset mật khẩu nhập vào  isEnteringPassword = false; // kết thúc quá trình nhập mật khẩu  }  }  }  prevSwitchState = switchState;  }  String getPassword() {  String payload = "";  if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi  HTTPClient http;  String url = String(FIREBASE\_HOST) + "/PASSWORD.json?auth=" + FIREBASE\_AUTH;  http.begin(url.c\_str());  int httpCode = http.GET();  if (httpCode > 0) {  payload = http.getString();  Serial.println(payload);  } else {  Serial.printf("GET request failed, error: %s\n", http.errorToString(httpCode).c\_str());  }  http.end();  } else {  Serial.println("WiFi not connected");  }  return payload;  }  String getPasswordTmp() {  String payload = "";  if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi  HTTPClient http;  String url = String(FIREBASE\_HOST) + "/TMP\_PASSWORD.json?auth=" + FIREBASE\_AUTH;  http.begin(url.c\_str());  int httpCode = http.GET();  if (httpCode > 0) {  payload = http.getString();  Serial.println(payload);  } else {  Serial.printf("GET request failed, error: %s\n", http.errorToString(httpCode).c\_str());  }  http.end();  } else {  Serial.println("WiFi not connected");  }  return payload;  }  void updateLEDStatus(const char\* status) {  if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi  HTTPClient http;  String url = String(FIREBASE\_HOST) + "/LED.json?auth=" + FIREBASE\_AUTH;  http.begin(url.c\_str());  http.addHeader("Content-Type", "application/json");  int httpCode = http.PUT(String("\"") + status + "\"");  if (httpCode > 0) {  String payload = http.getString();  Serial.println(payload);  } else {  Serial.printf("PUT request failed, error: %s\n", http.errorToString(httpCode).c\_str());  }  http.end();  } else {  Serial.println("WiFi not connected");  }  }  int getLEDStatus() {  if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi  HTTPClient http;  String url = String(FIREBASE\_HOST) + "/LED.json?auth=" + FIREBASE\_AUTH;  http.begin(url.c\_str());  int httpCode = http.GET();  if (httpCode > 0) {  String payload = http.getString();  Serial.println(payload);  if (payload == "\"ON\"") {  // digitalWrite(ledPin, HIGH);  Serial.println("LED is ON");  return 1;  } else if (payload == "\"OFF\"") {  // digitalWrite(ledPin, LOW);  Serial.println("LED is OFF");  return 0;  }  } else {  Serial.printf("GET request failed, error: %s\n", http.errorToString(httpCode).c\_str());  }  http.end();  } else {  Serial.println("WiFi not connected");  }  return 0;  }  String getTime() {  String payload = "";  if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi  HTTPClient http;  String url = String(FIREBASE\_HOST) + "/TIME.json?auth=" + FIREBASE\_AUTH;  http.begin(url.c\_str());  int httpCode = http.GET();  if (httpCode > 0) {  payload = http.getString();  Serial.println(payload);  } else {  Serial.printf("GET request failed, error: %s\n", http.errorToString(httpCode).c\_str());  }  http.end();  } else {  Serial.println("WiFi not connected");  }  return payload;  }  // void handleTime() {  // String data = getTime();  // timeClient.update(); // Cập nhật thời gian  // // Lấy thời gian từ NTP và gán cho các biến  // unsigned long epochTime = timeClient.getEpochTime(); // Lấy thời gian epoch  // // Cập nhật thời gian cho thư viện TimeLib  // setTime(epochTime);  // int dayCurrent = String(day()).toInt();  // int monthCurrent = String(month()).toInt();  // int yearCurrent = String(year()).toInt();  // int hourCurrent = String(hour()).toInt();  // int minuteCurrent = String(minute()).toInt();  // int day = data.substring(1,3).toInt();  // int month = data.substring(4,6).toInt();  // int year = data.substring(7,11).toInt();  // int hour = data.substring(12,14).toInt();  // int minute = data.substring(15).toInt();  // Serial.println(day);  // Serial.println(month);  // Serial.println(year);  // Serial.println(hour);  // Serial.println(minute);  // } |
| --- |

**Mã nguồn module thực hiện lệnh điều khiển:**

| **#include <ESP32Servo.h>**  **#include <WiFi.h>**  **#include <HTTPClient.h>**  **#include <ArduinoJson.h>**  **// wifi**  **const char\* ssid = "Duong";**  **const char\* passwordwf = "11111111";**  **// Thông tin Firebase**  **const char\* FIREBASE\_HOST = "https://bt-iot-7af8a-default-rtdb.asia-southeast1.firebasedatabase.app/";**  **const char\* FIREBASE\_AUTH = "i0SPueav25FRwD0lKD7BiQjGE9TVL9lVyhcuoDid";**  **Servo myServo; // Khai báo đối tượng Servo**  **void setup() {**  **Serial.begin(115200);**  **Serial.println("test");**  **myServo.attach(13); // Gắn servo vào chân GPIO 13 (hoặc chân khác tùy vào kết nối của bạn)**  **WiFi.begin(ssid, passwordwf);**  **while (WiFi.status() != WL\_CONNECTED) {**  **delay(1000);**  **Serial.println("Connecting to WiFi...");**  **}**  **Serial.println("Connected to WiFi");**  **}**  **int prevStatus = -1;**  **void loop() {**  **int status = getLEDStatus();**  **if(prevStatus==-1){**  **prevStatus = status;**  **}**  **if(status != prevStatus){**  **if(status == 1){**  **myServo.write(0);**  **}else if(status == 0){**  **myServo.write(180);**  **}**  **Serial.println(status);**  **prevStatus = status;**  **}**  **}**  **int getLEDStatus() {**  **if (WiFi.status() == WL\_CONNECTED) { // Kiểm tra kết nối WiFi**  **HTTPClient http;**  **String url = String(FIREBASE\_HOST) + "/LED.json?auth=" + FIREBASE\_AUTH;**  **http.begin(url.c\_str());**  **int httpCode = http.GET();**  **if (httpCode > 0) {**  **String payload = http.getString();**  **Serial.println(payload);**  **if (payload == "\"ON\"") {**  **// digitalWrite(ledPin, HIGH);**  **Serial.println("LED is ON");**  **return 1;**  **} else if (payload == "\"OFF\"") {**  **// digitalWrite(ledPin, LOW);**  **Serial.println("LED is OFF");**  **return 0;**  **}**  **} else {**  **Serial.printf("GET request failed, error: %s\n", http.errorToString(httpCode).c\_str());**  **}**  **http.end();**  **} else {**  **Serial.println("WiFi not connected");**  **}**  **return 0;**  **}** |
| --- |

**Mã nguồn ESP32-CAM thực hiện chụp ảnh**

| **#include <WebServer.h>**  **#include <WiFi.h>**  **#include <esp32cam.h>**  **const char\* WIFI\_SSID = "Duong";**  **const char\* WIFI\_PASS = "11111111";**  **// const char\* WIFI\_SSID = "TP-Link\_064A";**  **// const char\* WIFI\_PASS = "16755109";**    **WebServer server(80);**      **static auto loRes = esp32cam::Resolution::find(320, 240);**  **static auto midRes = esp32cam::Resolution::find(350, 530);**  **static auto hiRes = esp32cam::Resolution::find(800, 600);**  **void serveJpg()**  **{**  **auto frame = esp32cam::capture();**  **if (frame == nullptr) {**  **Serial.println("CAPTURE FAIL");**  **server.send(503, "", "");**  **return;**  **}**  **Serial.printf("CAPTURE OK %dx%d %dbn", frame->getWidth(), frame->getHeight(),**  **static\_cast<int>(frame->size()));**    **server.setContentLength(frame->size());**  **server.send(200, "image/jpeg");**  **WiFiClient client = server.client();**  **frame->writeTo(client);**  **}**    **void handleJpgLo()**  **{**  **if (!esp32cam::Camera.changeResolution(loRes)) {**  **Serial.println("SET-LO-RES FAIL");**  **}**  **serveJpg();**  **}**    **void handleJpgHi()**  **{**  **if (!esp32cam::Camera.changeResolution(hiRes)) {**  **Serial.println("SET-HI-RES FAIL");**  **}**  **serveJpg();**  **}**    **void handleJpgMid()**  **{**  **if (!esp32cam::Camera.changeResolution(midRes)) {**  **Serial.println("SET-MID-RES FAIL");**  **}**  **serveJpg();**  **}**      **void setup(){**  **Serial.begin(115200);**  **Serial.println();**  **{**  **using namespace esp32cam;**  **Config cfg;**  **cfg.setPins(pins::AiThinker);**  **cfg.setResolution(hiRes);**  **cfg.setBufferCount(2);**  **cfg.setJpeg(80);**    **bool ok = Camera.begin(cfg);**  **Serial.println(ok ? "CAMERA OK" : "CAMERA FAIL");**  **}**  **WiFi.persistent(false);**  **WiFi.mode(WIFI\_STA);**  **WiFi.begin(WIFI\_SSID, WIFI\_PASS);**  **while (WiFi.status() != WL\_CONNECTED) {**  **delay(500);**  **}**  **Serial.print("http://");**  **Serial.println(WiFi.localIP());**  **Serial.println(" /cam-lo.jpg");**  **Serial.println(" /cam-hi.jpg");**  **Serial.println(" /cam-mid.jpg");**    **server.on("/cam-lo.jpg", handleJpgLo);**  **server.on("/cam-hi.jpg", handleJpgHi);**  **server.on("/cam-mid.jpg", handleJpgMid);**    **server.begin();**  **}**    **void loop()**  **{**  **server.handleClient();**  **}** |
| --- |

## 2.4 Giải thích tổng quan về mã nguồn

**Cách Hoạt Động**

1. **Khởi Tạo**: Chương trình bắt đầu bằng việc khởi tạo các thư viện cần thiết và định nghĩa các chân kết nối cho cảm biến, động cơ, và các nút bấm. Các thông tin về mạng WiFi và Firebase cũng được thiết lập tại đây.
2. **Kết Nối WiFi**: Trong hàm setup(), chương trình sẽ kết nối với WiFi bằng SSID và mật khẩu đã được định nghĩa. Nếu kết nối thành công, nó sẽ thông báo trên Serial Monitor.
3. **Vòng Lặp Chính**: Trong vòng lặp loop(), chương trình sẽ thực hiện các bước sau:
   * Gọi hàm getSensorValues() để lấy giá trị cảm biến từ Firebase.
   * Kiểm tra trạng thái của nút bấm và cảm biến hành trình. Nếu nút bấm được nhấn, LED sẽ bật lên nếu một trong hai cảm biến hành trình bị kích hoạt.
   * Kiểm tra trạng thái LED từ Firebase. Nếu trạng thái thay đổi, nó sẽ bật hoặc tắt LED và động cơ tương ứng.
   * Đọc giá trị ánh sáng từ LDR hoặc nhiệt độ từ DHT11 để điều khiển LED hoặc động cơ dựa trên các ngưỡng đã được xác định.
4. **Điều Khiển Động Cơ**: Khi hàm on() được gọi, động cơ sẽ quay theo một hướng cho đến khi cảm biến hành trình thứ hai được kích hoạt. Ngược lại, khi hàm off() được gọi, động cơ sẽ quay theo hướng ngược lại cho đến khi cảm biến hành trình đầu tiên được kích hoạt.
5. **Giao Tiếp với Firebase**: Các hàm như getSensorValues(), getLEDStatus(), và updateLEDStatus() thực hiện giao tiếp với Firebase để nhận hoặc cập nhật trạng thái của các cảm biến và thiết bị.

# 

# Chương 3 Hệ thống phần mềm

## 3.1 Phiên bản app trên điện thoại

### 3.1.1 Mục đích

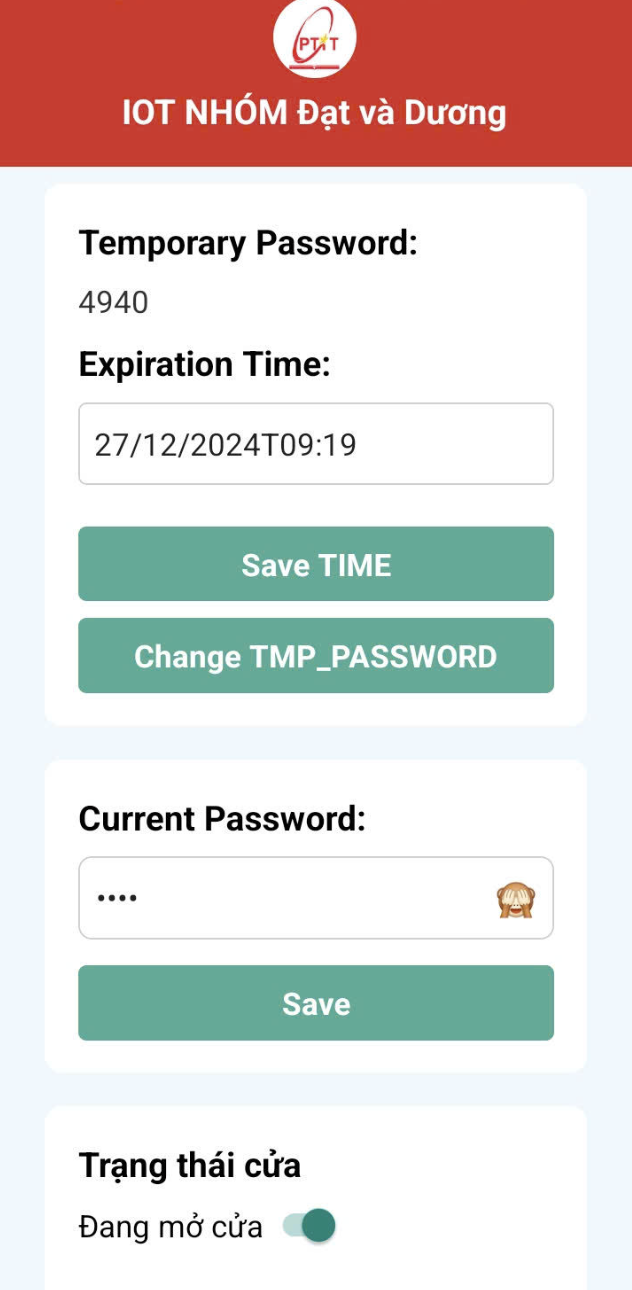
* Tăng cường sự tiện lợi cho người dùng: cho phép người dùng điều khiển cửa từ bất kỳ đâu thông qua kết nối internet, giúp tối ưu hóa thời gian và mang lại sự tiện lợi.
* Hiện tại mới chỉ có phiên bản chạy trên Android, chưa hỗ trợ các hệ điều hành khác

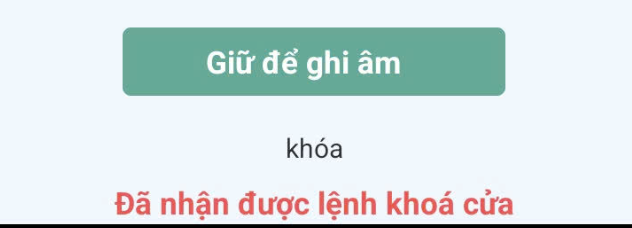
### 3.1.2 Công nghệ sử dụng

* **React Native**:
  + React Native là framework chủ yếu được sử dụng để phát triển ứng dụng di động. Với khả năng viết mã nguồn chung cho cả Android và iOS, React Native giúp tiết kiệm thời gian và công sức trong quá trình phát triển.
  + Ngoài ra, nó cung cấp các thành phần UI tái sử dụng, giúp nâng cao hiệu suất của ứng dụng và cải thiện trải nghiệm người dùng với giao diện mượt mà và thân thiện.
* **Expo**:
  + Expo là công cụ phát triển mạnh mẽ đi kèm với React Native, cung cấp các thư viện và API tích hợp sẵn để đơn giản hóa quy trình phát triển ứng dụng. Việc sử dụng Expo giúp các lập trình viên dễ dàng quản lý dự án mà không cần cấu hình phức tạp.
  + Hơn nữa, Expo hỗ trợ việc chạy thử nghiệm và triển khai ứng dụng nhanh chóng, từ đó giảm thời gian cần thiết để đưa ứng dụng đến tay người dùng.
* **Chuyển âm thanh sang văn bản với thư viện react-native-voice**:
  + Thư viện react-native-voice được sử dụng để tích hợp tính năng nhận diện giọng nói, cho phép người dùng điều khiển cửa chỉ bằng lệnh thoại. Điều này giúp tạo ra một trải nghiệm tương tác tự nhiên hơn cho người dùng.
  + Ngoài ra, thư viện này có khả năng xử lý nhiều ngôn ngữ, tạo điều kiện cho người dùng từ nhiều vùng miền có thể sử dụng ứng dụng dễ dàng và hiệu quả.
* **Deploy với APK**:
  + Ứng dụng được triển khai dưới dạng file APK, cho phép người dùng tải xuống và cài đặt trực tiếp trên thiết bị Android. Phương pháp này giúp đơn giản hóa quá trình phân phối ứng dụng mà không cần thông qua các cửa hàng ứng dụng.
  + Việc phát hành ứng dụng dưới dạng APK cũng cho phép nhóm phát triển dễ dàng kiểm soát phiên bản, thực hiện các cập nhật và sửa lỗi nhanh chóng mà không gặp phải những yêu cầu phức tạp của các nền tảng phân phối khác.

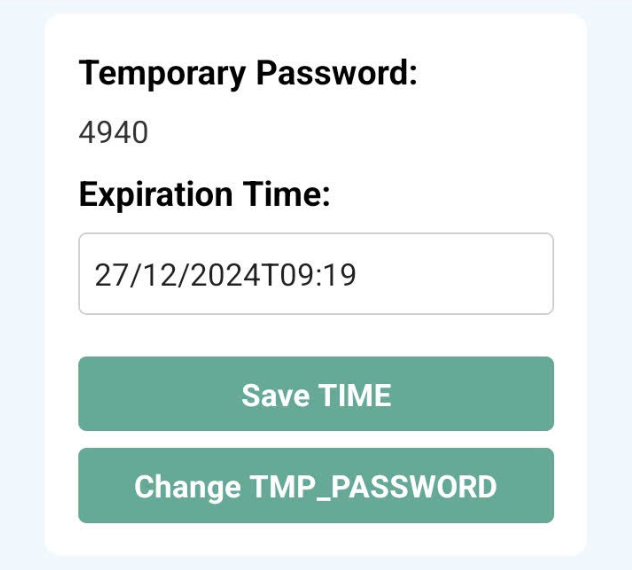
### 3.1.3 Các tính năng

* Giao diện của app

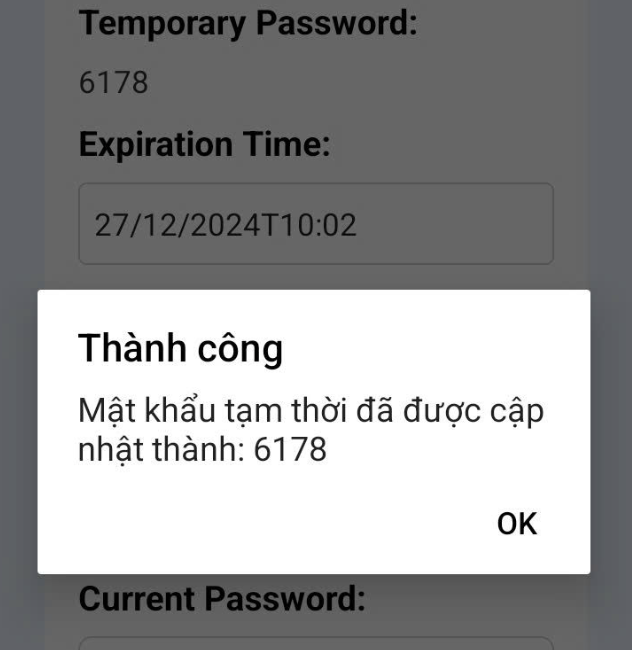




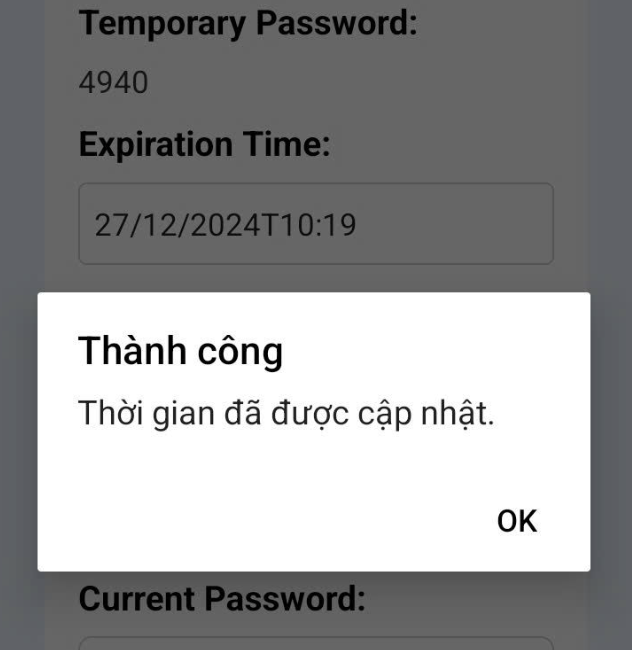
* Giao diện phần mật khẩu tạm thời



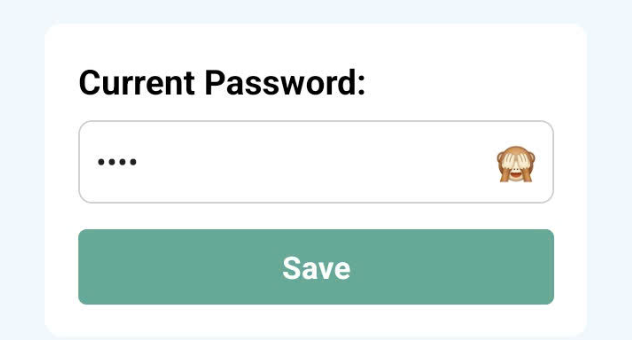
* Cung cấp mật khẩu tạm thời mỗi khi nhấn nút “Change TMP\_PASSWORD” và tự động đặt thời gian hết hạn sau 24 giờ kể từ khi nhấn nút



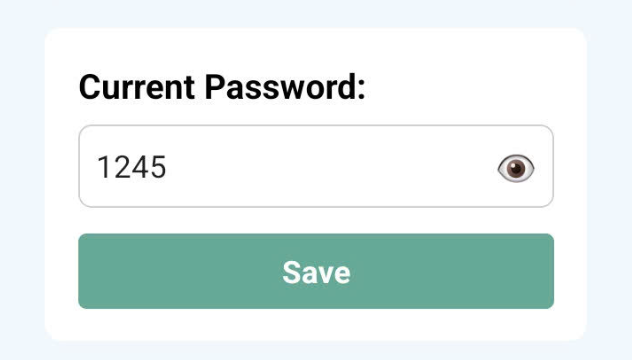
* Có thể thay đổi thời gian hết hạn bằng cách nhấn nút “Save TIME”



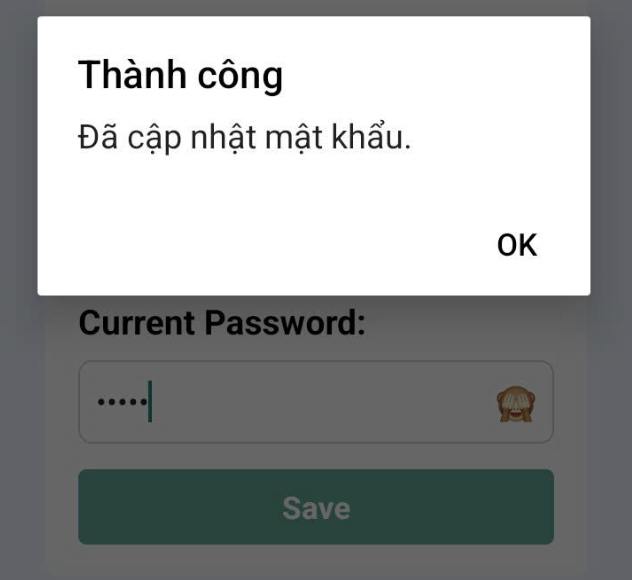
* Phần mật khẩu cố định



* Có thể hiển thị mật khẩu bằng cách nhấn vào Icon bên phải

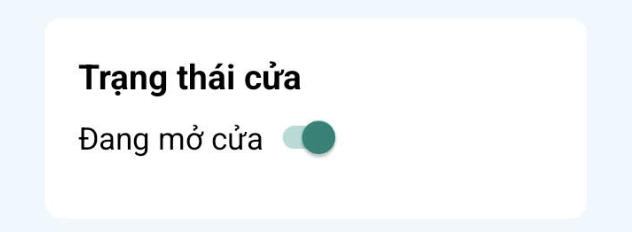


* Có thể thay đổi mật khẩu và nhấn “Save” để lưu

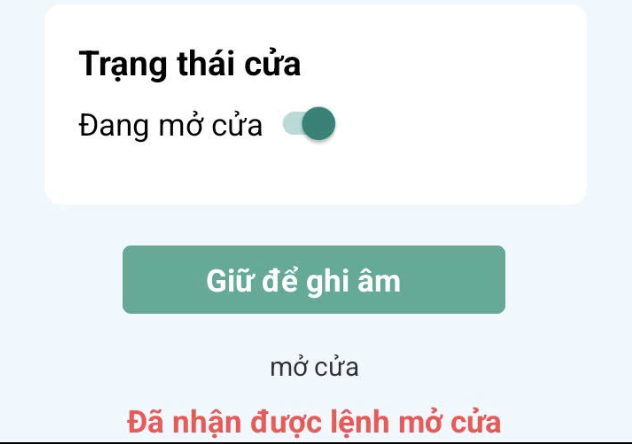


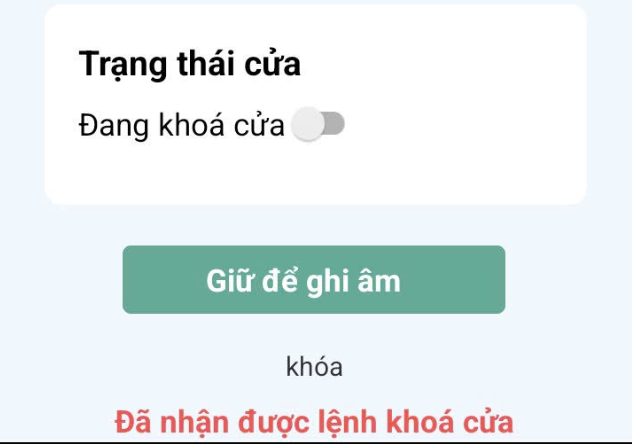
* Phần khoá mở cửa bằng nút bấm





* Phần khoá mở cửa bằng giọng nói





* + Người dùng nhấn giữ nút “Giữ để ghi âm” và nói lệnh, app sẽ chuyển âm thanh thành văn bản, nếu văn bản có chứa từ “khoá” hoặc “mở”, cửa sẽ tự động khoá hay mở theo lệnh.

## 

## 3.2 Phiên bản web

### 3.2.1 Mục đích

* Phiên bản web có thể chạy trên mọi hệ điều hành (Android, IOS, Window, MacOS, Linix, …)
* Tuy nhiên website chỉ hoạt động tốt trên trình duyệt Chrome do Chrome hỗ trợ tính năng chuyển âm thanh sang văn bản với **Web Speech API**

### 3.2.2 Công nghệ sử dụng

Để phát triển và triển khai phiên bản web của ứng dụng điều khiển mở khoá cửa, dự án sử dụng các công nghệ dưới đây nhằm tối ưu hóa hiệu năng, khả năng truy cập đa nền tảng và mang lại trải nghiệm nhất quán cho người dùng:

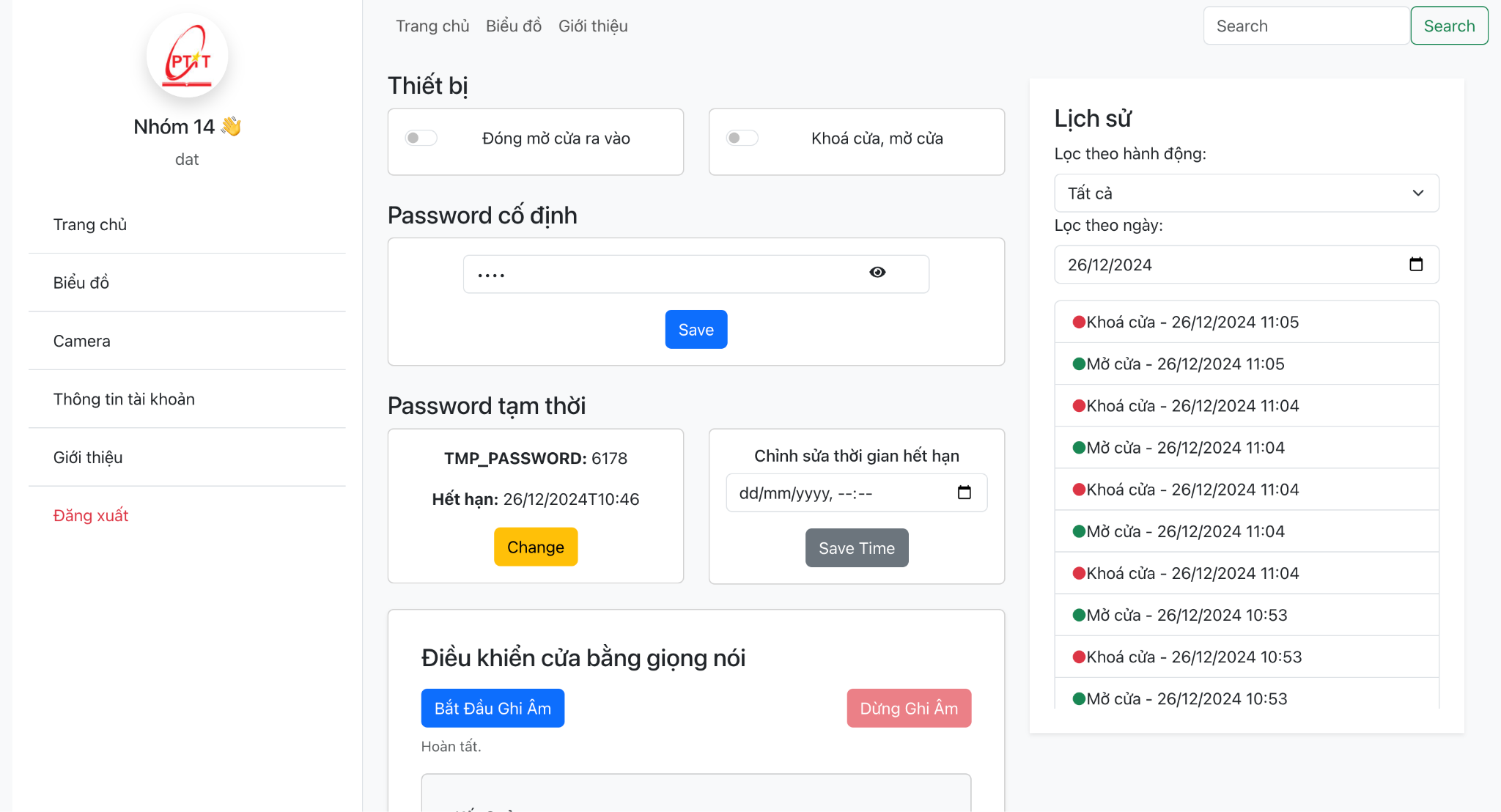
* **ReactJS**:
  + ReactJS là framework chính để xây dựng giao diện người dùng cho phiên bản web. Với khả năng xây dựng các thành phần UI (User Interface) có thể tái sử dụng, ReactJS giúp tạo ra giao diện linh hoạt, tối ưu hóa tốc độ phản hồi và tăng tính thân thiện cho người dùng.
  + ReactJS cũng dễ dàng kết hợp với các thư viện và công cụ khác, giúp mở rộng tính năng và khả năng tương tác cho ứng dụng, phù hợp với yêu cầu điều khiển và giám sát khoá cửa từ xa.
* **Chuyển âm thanh sang văn bản với Web Speech API**:
  + **Web Speech API** của Chrome được sử dụng để tích hợp tính năng nhận diện giọng nói, cho phép người dùng điều khiển khoá cửa thông qua lệnh thoại. Tính năng này giúp mở rộng tiện ích và tăng cường sự tương tác tự nhiên với ứng dụng.
  + Với Web Speech API, trình duyệt có thể nhận diện giọng nói, chuyển đổi sang văn bản và gửi lệnh đến hệ thống điều khiển cửa, mang lại sự tiện lợi và trải nghiệm liền mạch.
  + Do hiện tại Web Speech API chỉ hỗ trợ tốt trên Chrome, ứng dụng web được tối ưu hóa để chạy trên trình duyệt này.
* **Triển khai với Vercel**:
  + **Vercel** là nền tảng triển khai (deployment platform) được sử dụng để đưa ứng dụng web vào môi trường trực tuyến, giúp người dùng có thể truy cập ứng dụng từ bất kỳ thiết bị và hệ điều hành nào.
  + Với khả năng CI/CD tích hợp, Vercel cho phép các bản cập nhật và tính năng mới được triển khai tự động, đảm bảo phiên bản ứng dụng luôn là phiên bản mới nhất.
  + Ngoài ra, Vercel cung cấp tốc độ tải trang nhanh và ổn định, tối ưu hóa cho các ứng dụng React, giúp nâng cao trải nghiệm người dùng trên phiên bản web.
  + Có thể truy cập web trên Chrome qua link <https://iot-indol-one.vercel.app/login> (Tài khoản: dat, mật khẩu: 123)

### 3.2.3 Các tính năng

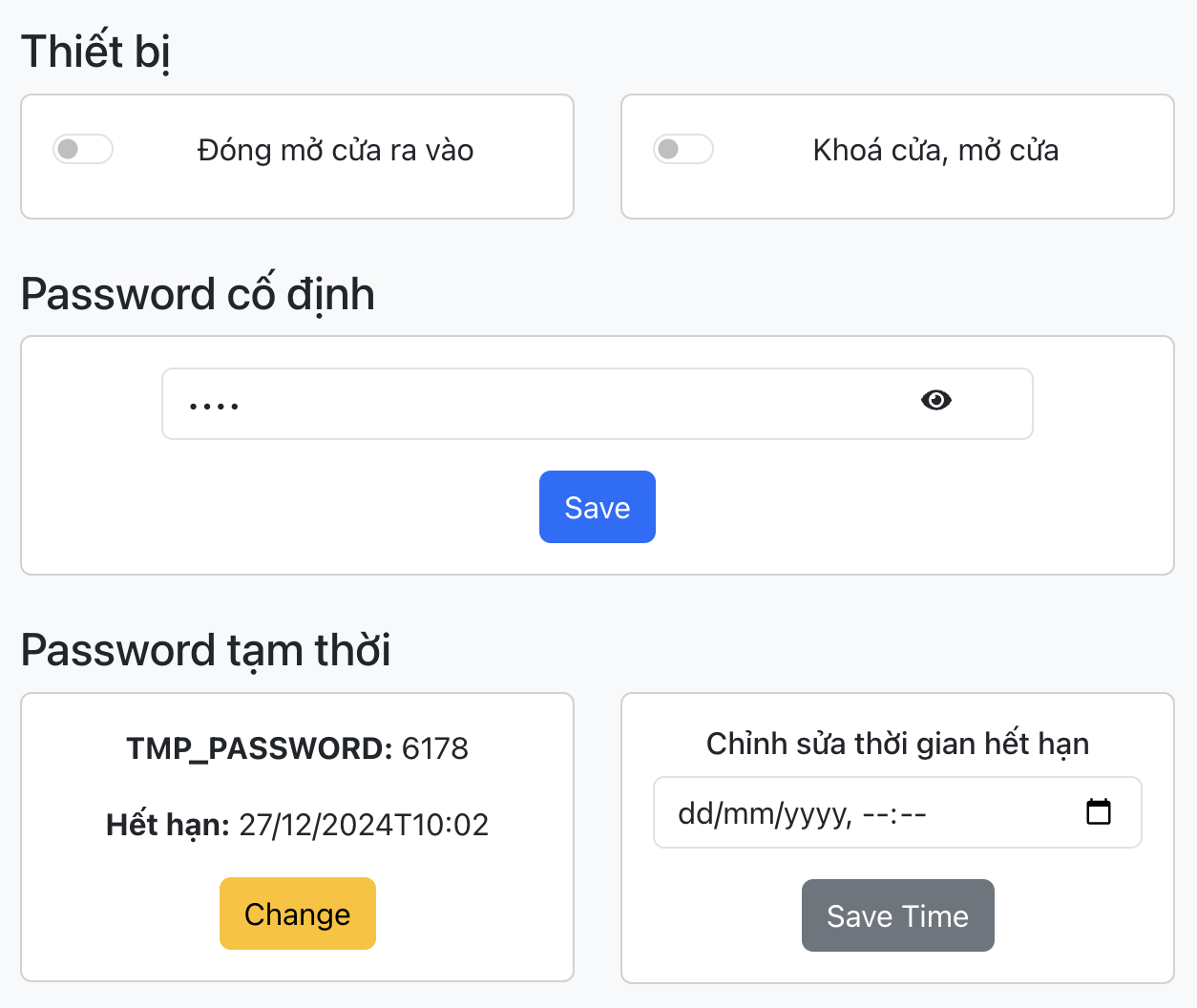
* Đăng nhập: để tăng khả năng bảo mật, web yêu cầu đăng nhập trước khi truy cập vào trang chủ



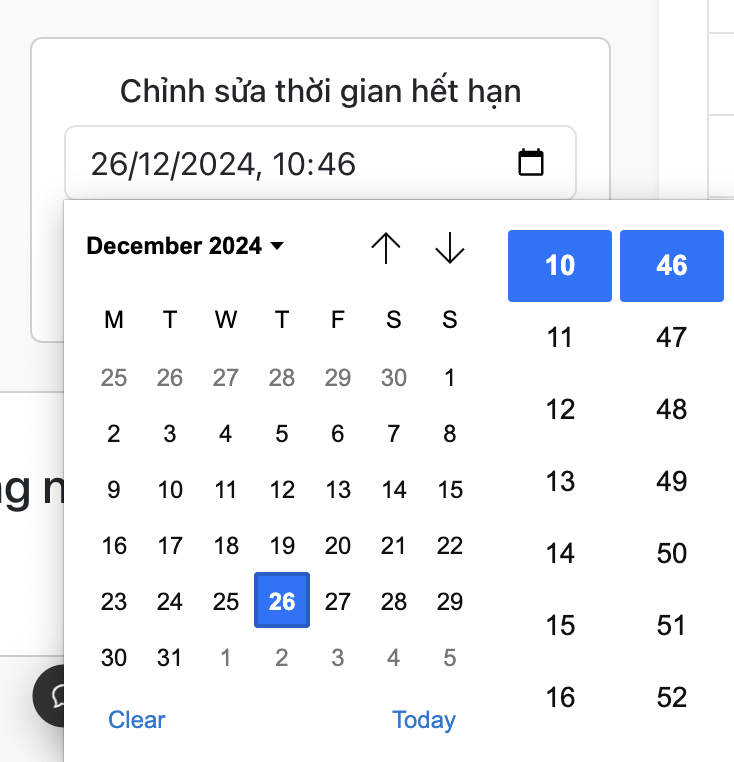
* Sau khi đăng nhập thành công sẽ chuyển hướng đến trang chủ



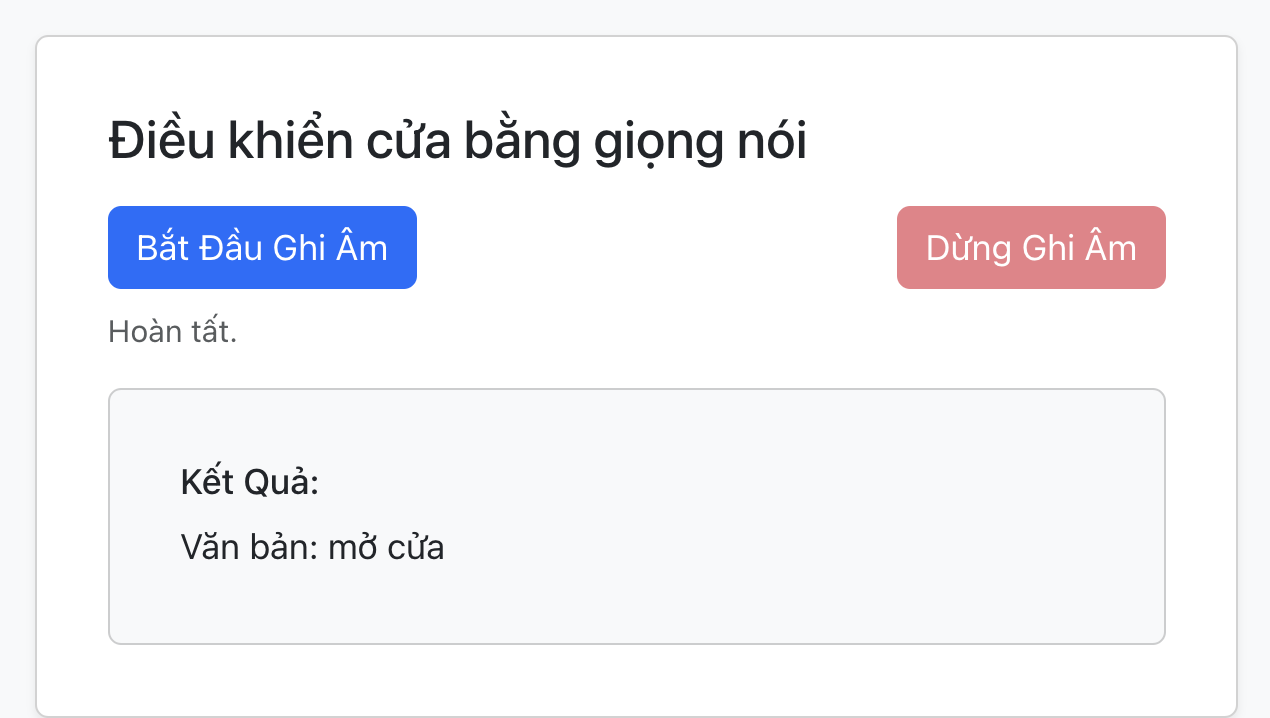
* Điều khiển khóa mở cửa và password

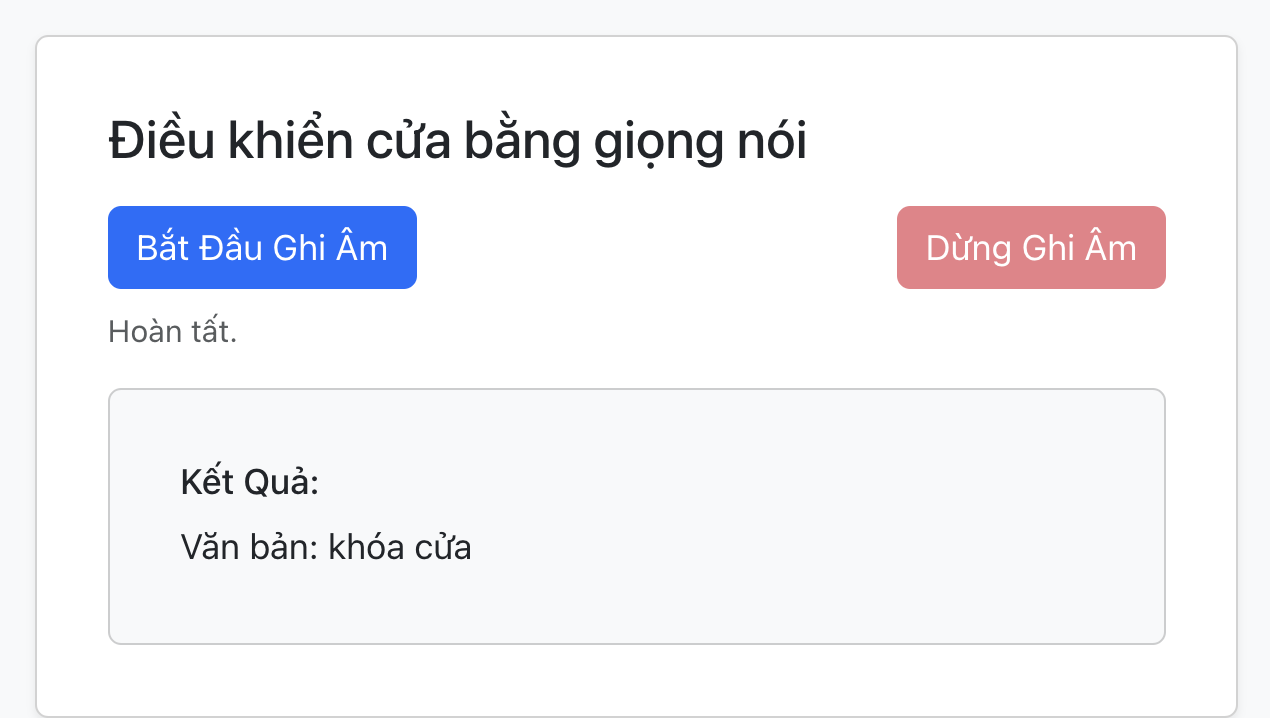


* + Để khoá mở từ xa, người dùng nhấn nút “Khoá cửa, mở cửa” ở phần thiết bị
  + Có thể hiển thị và thay đổi mật khẩu cố định bằng cách nhấn nút Save
  + Có thể thay đổi mật khẩu tạm thời bằng cách nhấn nút Change, thay đổi thời gian hết hạn bằng cách chọn thời gian và nhấn nút Save TIme

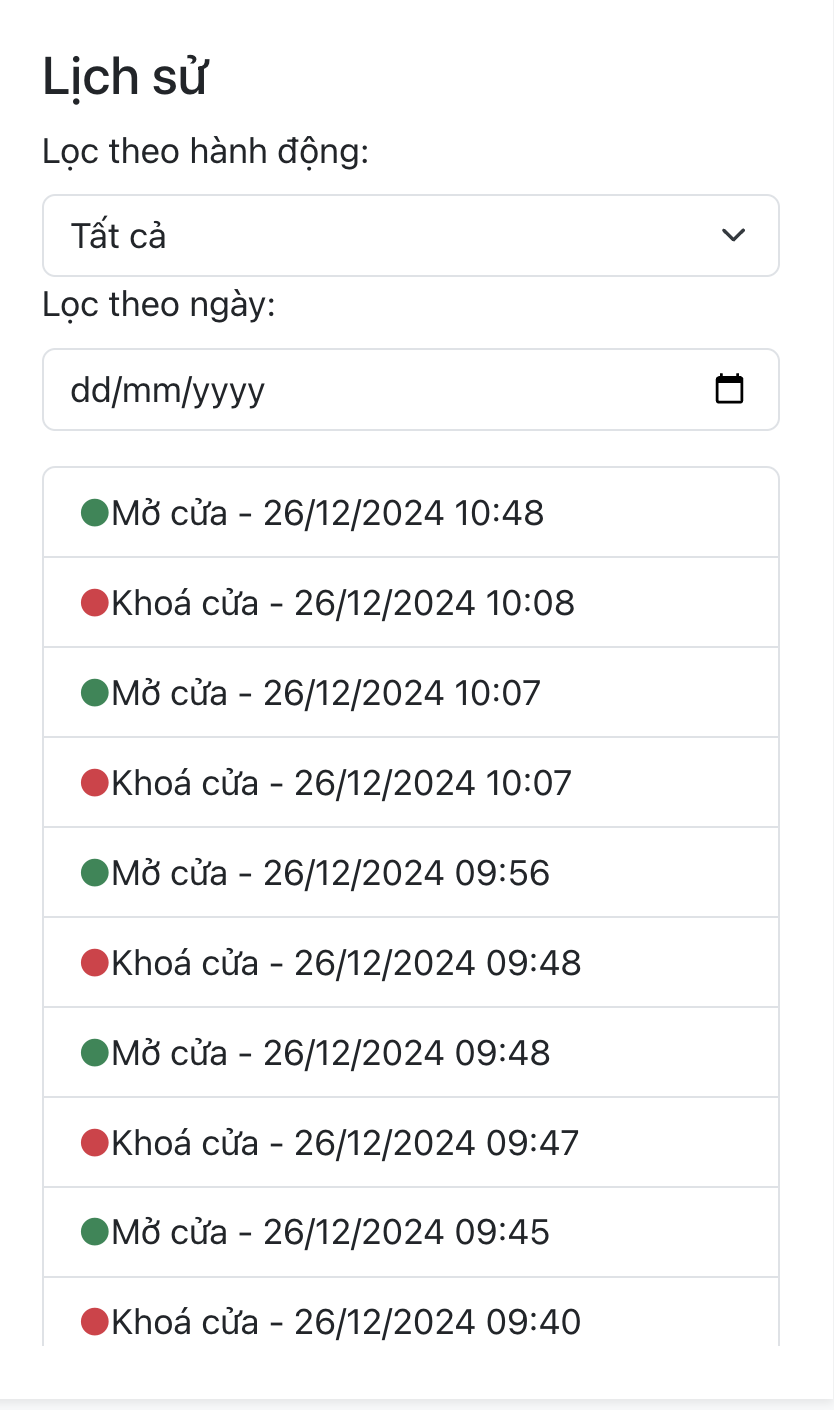


* Điều khiển khoá mở cửa dùng giọng nói





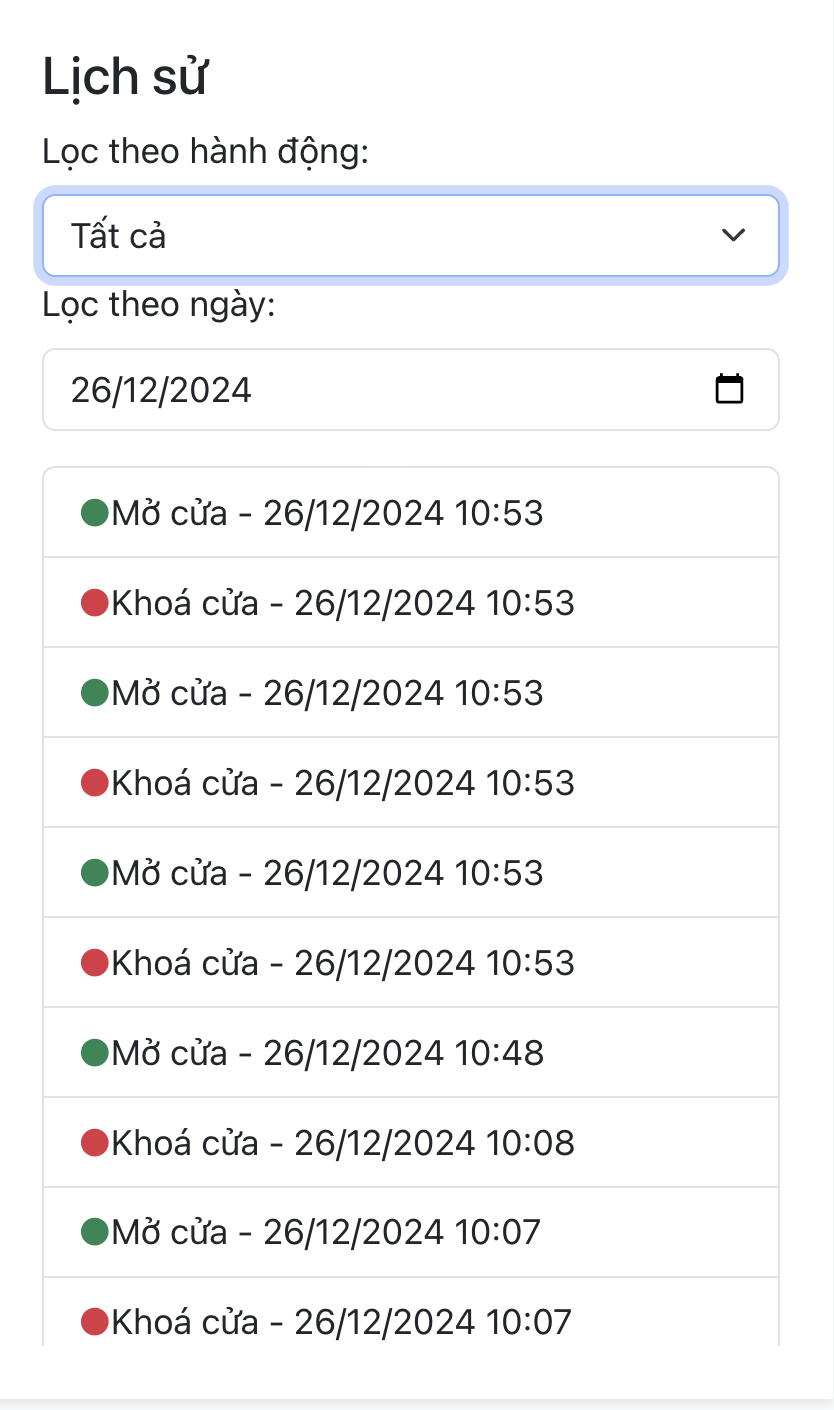
* Người dùng bấm nút bắt đầu ghi, sau đó nói lệnh và nhấn dừng ghi âm, **Web Speech API** sẽ chuyển giọng nói thành văn bản, nếu văn bản là “mở cửa” hoặc “khoá cửa”, thao tác tương ứng sẽ được thưc hiện
* Xem lịch sử khoá mở cửa



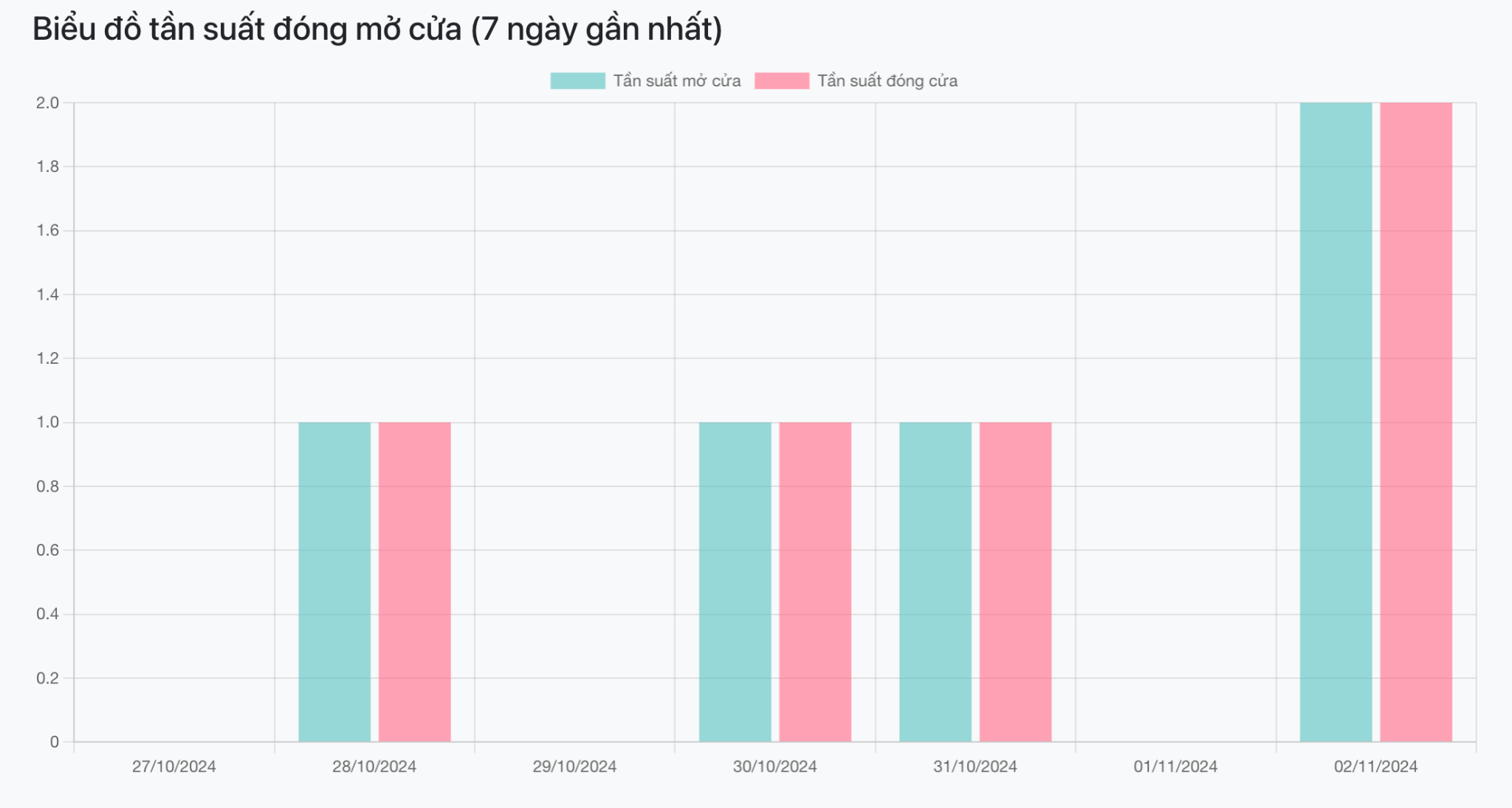
* Người dùng có thể lọc lịch sử theo hành động

|  |  |
| --- | --- |

* Người dùng cũng có thể lọc lịch sử theo từng ngày



* Xem biểu đồ tần suất khoá mở cửa



# 

# Chương 4 Kết quả

## 4.1. Chức năng đã hoàn thành

* **Điều khiển từ xa:** Khóa cửa thông minh cho phép mở hoặc khóa cửa từ xa qua ứng dụng di động, rất tiện lợi khi quên chìa khóa hoặc cần cho phép ai đó vào nhà. Ví dụ, bạn có thể mở cửa cho khách qua điện thoại mà không cần chìa khóa vật lý.
* **Mã số và vân tay:** Khóa thông minh hỗ trợ mở bằng mã số hoặc vân tay, tăng cường bảo mật và tiện lợi hơn chìa khóa truyền thống. Các mã số được quản lý an toàn qua ứng dụng trực tuyến.
* **Kết nối ứng dụng:** Khóa tích hợp ứng dụng di động giúp mở cửa, kiểm tra trạng thái, nhận thông báo về hoạt động bất thường hoặc truy cập trái phép, nâng cao kiểm soát và bảo mật.
* **Lịch sử truy cập:** Lưu trữ thông tin về thời gian và người truy cập, hỗ trợ theo dõi ra vào và quản lý quyền truy cập, hữu ích trong gia đình, văn phòng hoặc khách sạn.
* **Khóa tự động:** Tự động khóa khi cửa đóng, đảm bảo an toàn nếu quên khóa. Một số khóa còn mở tự động khi nhận diện thiết bị di động, tăng sự tiện lợi.
* **Mã pin tạm thời:** Tạo mã pin giới hạn thời gian cho khách, đảm bảo họ chỉ mở cửa trong khoảng thời gian được phép.

## 4.2. Hiệu suất hệ thống

**Đánh giá về tốc độ phản hồi**a) **Thời gian phản hồi của API**:

Các API endpoint cung cấp dữ liệu cảm biến và cho phép điều khiển khoá có thời gian phản hồi trung bình dưới 200ms cho các truy vấn cơ bản.

Với các truy vấn phức tạp hơn, như xem lịch sử điều khiển hoặc điều khiển nhiều thiết bị cùng lúc, thời gian phản hồi vẫn duy trì dưới 500ms.

b) **Độ trễ khi điều khiển từ xa**:

Thời gian từ khi người dùng gửi lệnh điều khiển trên ứng dụng đến khi rèm thực hiện thay đổi trung bình là 0.5 - 1 giây.

Độ trễ chủ yếu do thời gian truyền dữ liệu qua MQTT và xử lý trên ESP32, nhưng hệ thống vẫn đảm bảo đáp ứng đủ nhanh cho các nhu cầu sử dụng thông thường.

## 4.3. Cải tiến trong tương lai

* **Tích hợp AI:** Ứng dụng trí tuệ nhân tạo để nhận diện hành vi bất thường, phát hiện các mối đe dọa tiềm ẩn hoặc tự động đề xuất nâng cấp bảo mật dựa trên thói quen sử dụng.
* **Kết nối đa thiết bị:** Hỗ trợ tích hợp sâu với các hệ thống nhà thông minh khác, như camera an ninh, cảm biến chuyển động hoặc loa thông minh, để tạo trải nghiệm liền mạch hơn.
* **Pin năng lượng tái tạo:** Trang bị pin năng lượng mặt trời hoặc cơ chế tự sạc từ năng lượng cơ học khi sử dụng, giúp tăng thời gian hoạt động và bảo vệ môi trường.
* **Chế độ khẩn cấp:** Phát triển chế độ khẩn cấp cho phép khóa cửa tự động hoặc gửi tín hiệu SOS đến người thân hoặc cơ quan chức năng khi phát hiện nguy cơ xâm nhập.
* **Cập nhật không dây:** Tự động cập nhật firmware để bổ sung tính năng mới hoặc vá lỗi bảo mật, đảm bảo hệ thống luôn an toàn và tối ưu.
* **Khả năng mở khóa không tiếp xúc:** Áp dụng công nghệ giao tiếp gần (NFC) hoặc sóng vô tuyến, cho phép người dùng mở khóa mà không cần chạm vào bất kỳ thiết bị nào.
* **Cá nhân hóa cao cấp:** Cho phép tùy chỉnh giao diện và tính năng theo nhu cầu người dùng, như chọn ngôn ngữ, giọng nói thông báo hoặc cài đặt chế độ riêng tư theo thời gian thực.

# 

# Lời cảm ơn

Chúng em xin gửi lời cảm ơn sâu sắc đến thầy Nguyễn Tài Tuyên, giảng viên môn học Internet of Things và Ứng dụng tại Học viện Công nghệ Bưu chính Viễn thông.

Trong suốt khóa học, thầy đã truyền đạt kiến thức một cách tận tâm và chuyên nghiệp, giúp em và các bạn sinh viên hiểu sâu sắc về lĩnh vực IoT đang phát triển nhanh chóng. Những bài giảng sinh động và các ví dụ thực tế của thầy đã truyền cảm hứng cho chúng em, khơi dậy niềm đam mê với công nghệ và mở ra nhiều cơ hội mới trong tương lai nghề nghiệp.

Thầy không chỉ là một người thầy giàu kinh nghiệm mà còn là một người cố vấn tận tụy. Sự hướng dẫn và ủng hộ của thầy trong quá trình thực hiện đồ án này đã giúp em vượt qua nhiều khó khăn, từ đó hoàn thiện dự án một cách tốt nhất.

Kiến thức và kỹ năng chúng em học được từ môn học này chắc chắn sẽ là nền tảng vững chắc cho sự phát triển nghề nghiệp của em trong tương lai. Chúng em xin chân thành cảm ơn thầy vì tất cả những đóng góp quý báu này.

Kính chúc thầy luôn mạnh khỏe, hạnh phúc và thành công trong sự nghiệp giảng dạy cao quý của mình.

Trân trọng.