

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



Thực tập đồ án đa ngành
Hướng công nghệ phần mềm (CO3109)

**SMART HOME - ỨNG DỤNG
QUẢN LÝ NHÀ THÔNG MINH**

GVHD:	Nguyễn Hữu Hiếu	
SV thực hiện:	Nguyễn Hữu Danh	2010174
	Trần Nguyên Vũ	2012445
	Nguyễn Huỳnh Tuấn Hưng	2011329
	Ché Lan Hải	2013063
	Lê Quang Tuấn Hào	2013048

Thành phố Hồ Chí Minh, tháng 4 năm 2023



Mục lục

I Requirement Elitication	4
1 Mô tả dự án	4
2 Yêu cầu chức năng	4
3 Yêu cầu phi chức năng	4
4 Use case diagram	6
4.1 Đăng nhập/ Đăng ký tài khoản	6
4.1.1 Use case diagram	6
4.1.2 Use case scenario - Đăng nhập	7
4.1.3 Use case scenario - Đăng ký	8
4.2 Chính sửa thông tin người dùng	9
4.2.1 Use case diagram	9
4.2.2 Use case scenario	10
4.3 Quản lý các phòng	11
4.3.1 Use case diagram	11
4.3.2 Use case scenario	12
4.4 Quản lý thông tin thiết bị	13
4.4.1 Use case diagram	13
4.4.2 Use case scenario	13
4.5 Hiển thị dữ liệu cảm biến, trạng thái thiết bị lên màn hình	15
4.5.1 Use case diagram	15
4.5.2 Use case scenario	15
4.6 Điều khiển thiết bị trong nhà	16
4.6.1 Use case diagram	16
4.6.2 Use case scenario	16
4.7 Thông báo khi nhiệt độ, độ ẩm vượt ngưỡng quy định	16
4.7.1 Use case diagram	16
4.7.2 Usecase scenario	17
4.8 Xem lịch sử ghi nhận sử dụng thiết bị và cảm biến	18
4.8.1 Use case diagram	18
4.8.2 Usecase scenario	18
II Implementation	20



5 Phần cứng	20
5.1 Các thiết bị sử dụng	20
5.1.1 Mạch lập trình	20
5.1.2 Các thiết bị ngoại vi	21
5.1.3 Kết nối các thiết bị	24
5.2 Các tính năng triển khai	24
5.2.1 Tổng quan tính năng	24
5.2.2 Các khối lệnh	25
5.3 Source Code	29
6 Phần mềm	30
6.1 Thiết kế cơ sở dữ liệu	30
6.2 Thiết kế UI	32
6.2.1 Trang đăng nhập, đăng ký và chỉnh sửa thông tin người dùng	32
6.2.2 Trang dashboard	33
6.2.3 Trang thiết bị và cảm biến	34
6.2.4 Giao diện xem thông tin phòng	35
6.2.5 Trang xem lịch sử	37
6.3 Tổ chức thư mục trong dự án	38
6.4 Kiến trúc hệ thống	40
6.5 Công nghệ được sử dụng	41
6.5.1 ReactJS	41
6.5.2 NodeJS	41
6.5.3 ExpressJS	42
6.5.4 MongoDB	43
6.5.5 Socket.IO	43
6.5.6 JSON Web Token	44
6.5.7 Formik	45
6.6 Sản phẩm hiện thực	46
6.6.1 Design Pattern	46
6.6.2 API documentation	47
6.6.3 Kết quả hiện thực trang web	48
6.6.4 Mã nguồn của nhóm	57



Phân chia công việc

Họ và Tên	Công việc	Mức độ hoàn thiện (%)
Nguyễn Hữu Danh	Chức năng quản lý phòng, lịch sử ghi nhận dữ liệu từ thiết bị, hiển thị dashboard cho từng phòng.	100%
Trần Nguyên Vũ	Chức năng đăng nhập, đăng ký, chỉnh sửa thông tin tài khoản, quản lý thiết bị	100%
Nguyễn Huỳnh Tuấn Hưng	Chức năng hiển thị dashboard cho từng phòng, làm các API để xử lý sự kiện bên giao diện, thiết kế database	100%
Chế Lan Hải	Kết nối phần cứng với hệ thống	100%
Lê Quang Tuấn Hào	Kết nối phần cứng với hệ thống	100%



Phần I

Requirement Elicitation

1 Mô tả dự án

Nhóm tạo một nền tảng web để giúp theo dõi thông tin của các thiết bị và cảm biến được cài đặt trên những căn phòng ảo theo thời gian thực. Các dữ liệu được gửi liên tục từ các cảm biến, dữ liệu sẽ được trực quan hóa dưới dạng bảng để người dùng có thể theo dõi dễ dàng.

Đồng thời các dữ liệu từ cảm biến sẽ được hiển thị thông qua dashboard của từng căn phòng và các bảng LED được gắn trong nhà để người dùng tiện lợi theo dõi. Hệ thống cũng cho phép người dùng điều khiển các thiết bị trong nhà như quạt, đèn, cũng như tự động bật đèn và mở cửa trong nhà thông qua cảm biến phát hiện chuyển động.

Hệ thống cho phép người dùng thiết lập nhiều căn phòng ảo khác nhau, trong căn phòng đó người dùng có thể kết nối với các thiết bị thông qua mã code được định danh riêng cho từng thiết bị đây.

Những thông số người dùng có thể theo dõi bao gồm nhiệt độ, độ ẩm, trạng thái bật/tắt của đèn và thông số của quạt. Các thông số này sẽ được cập nhật mỗi 10 phút trên các bảng đèn LED và trên database. Hệ thống sẽ gọi đến các thông số này để hiển thị trên giao diện người dùng thông qua các API. Khi các thông số này ra khỏi ngưỡng quy định do người dùng thiết lập, sẽ có thông báo được gửi đến người dùng trên giao diện hệ thống. Ngoài ra người dùng có thể cập nhật mô tả cho các thiết bị và cảm biến này để hệ thống cập nhật lên database.

2 Yêu cầu chức năng

- Đăng ký/Đăng nhập.
- Chỉnh sửa thông tin tài khoản.
- Tạo / Xóa / Sửa nhà cần quản lý.
- Thêm / Xóa / Chỉnh sửa thông tin thiết bị trong nhà.
- Hiển thị thông số cảm biến, trạng thái thiết bị lên màn hình.
- Điều khiển thiết bị (quạt, đèn, cửa).
- Thông báo khi nhiệt độ, độ ẩm vượt ngưỡng quy định.
- Xem lịch sử dữ liệu từ các cảm biến và số lần sử dụng thiết bị.

3 Yêu cầu phi chức năng

- Giao diện thân thiện và có thể hiểu ngay cách sử dụng sau khi xem qua hướng dẫn sử dụng 1 lần.

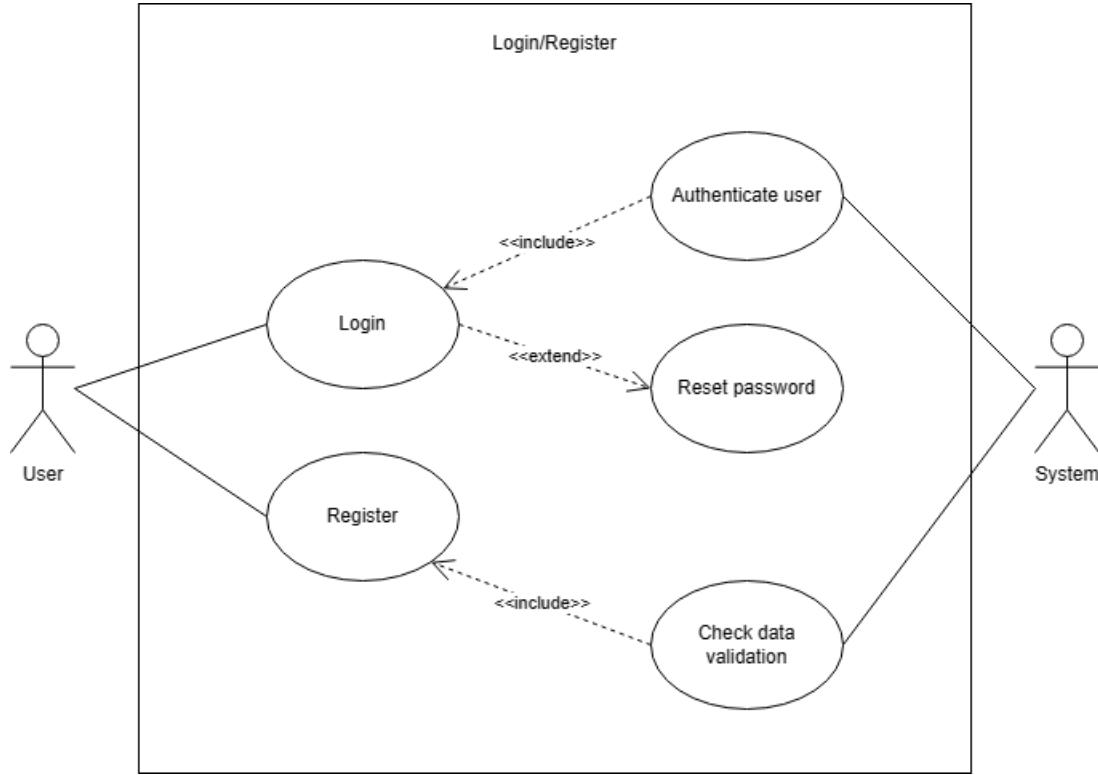


- Hệ thống có thể đáp ứng tối đa 100 người cùng lúc.
- Database có thể tiếp nhận dữ liệu liên tục từ 1000 cảm biến đồng thời.
- Các dữ liệu sẽ được cập nhật trên trang web mỗi 10 phút.
- Cảnh báo sẽ được gửi đến người dùng trong vòng 1s.
- Hỗ trợ 2 nền tảng web là Google Chrome và Firefox.
- Hệ thống sẽ phản hồi lại cho người dùng trong vòng 1s.

4 Use case diagram

4.1 Đăng nhập/ Đăng ký tài khoản

4.1.1 Use case diagram



Hình 1: Use case diagram cho tính năng đăng nhập và đăng ký



4.1.2 Use case scenario - Đăng nhập

Use case name	Đăng nhập tài khoản
Actor	Người dùng
Description	Tính năng đăng nhập cho phép người dùng đăng nhập vào hệ thống bằng tên đăng nhập và mật khẩu của họ. Nếu thông tin đăng nhập chính xác, họ có thể truy cập vào các tính năng được phân quyền.
Preconditions	Người dùng cần có tài khoản đã đăng ký trên hệ thống
Postconditions	Người dùng được đăng nhập vào hệ thống và có thể truy cập vào các tính năng được phân quyền
Trigger	Người dùng chọn tính năng đăng nhập trên giao diện người dùng
Main Flow	<ol style="list-style-type: none">Hệ thống hiển thị trang đăng nhập và yêu cầu người dùng nhập tên đăng nhập và mật khẩuNgười dùng nhập thông tin tài khoản của mình vào các trường tương ứngNgười dùng nhấn nút "Đăng nhập"Hệ thống xác minh thông tin đăng nhập và cho phép người dùng truy cập vào hệ thống nếu thông tin đăng nhập chính xác
Alternative Flow	<p>Nếu người dùng nhập sai thông tin đăng nhập, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng nhập lại</p> <ol style="list-style-type: none">Hệ thống hiện cảnh báo lỗi sai thông tin đăng nhậpNgười dùng nhập lại thông tin đăng nhập <p>Nếu hệ thống gặp lỗi khi xác minh thông tin đăng nhập, nó sẽ hiển thị thông báo lỗi và yêu cầu người dùng thử lại sau</p> <ol style="list-style-type: none">Hệ thống hiện lỗi khi xác minh thông tin đăng nhậpHệ thống thông báo yêu cầu người dùng thử lại sau
Exception Flows	<p>Nếu người dùng quên mật khẩu, hệ thống cung cấp tính năng lấy lại mật khẩu để người dùng khôi phục mật khẩu của mình</p> <ol style="list-style-type: none">Người dùng nhấn vào nút quên mật khẩu.Hệ thống gửi mã xác thực đến email người dùng.Người dùng nhập mã xác thực và đặt lại mật khẩu.Người dùng tiếp tục đăng nhập như bình thường

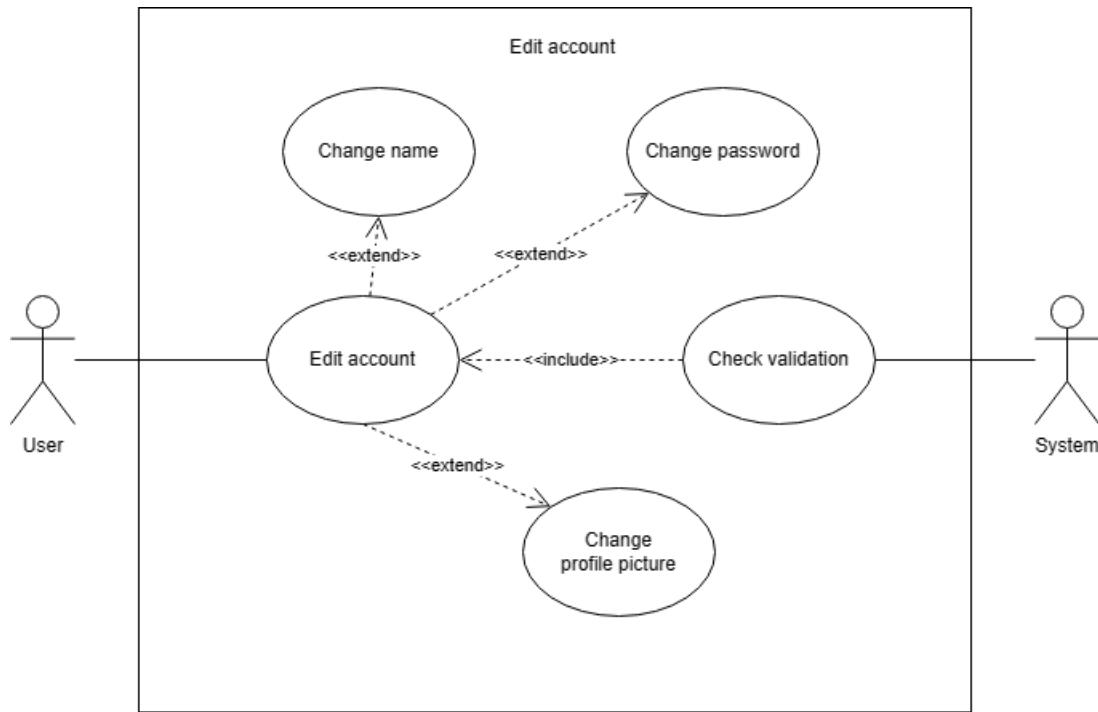


4.1.3 Use case scenario - Đăng ký

Use case name	Đăng ký tài khoản
Actor	Người dùng
Description	Tính năng ký cho phép người dùng tạo tài khoản với email và mật khẩu, sau đó người dùng có thể sử dụng tài khoản này để đăng nhập vào hệ thống của trang web.
Preconditions	Người dùng truy cập vào trang đăng ký tài khoản
Postconditions	Người dùng đã đăng ký tài khoản thành công và có thể sử dụng tài khoản đó để truy cập vào trang web
Trigger	Người dùng chọn tính năng đăng ký trên giao diện người dùng
Main Flow	<ol style="list-style-type: none">Người dùng truy cập vào trang web đăng kýNgười dùng nhấn vào nút "Đăng ký" và chuyển đến trang đăng kýNgười dùng nhập thông tin cá nhân của mình như tên, địa chỉ email, mật khẩu, số điện thoại,...Người dùng nhấn vào nút "Đăng ký" để hoàn tất quá trình đăng ký. Hệ thống kiểm tra thông tin đăng ký của người dùng và tạo tài khoản mới trong cơ sở dữ liệuHệ thống chuyển người dùng đến trang thông báo đăng ký thành công
Alternative Flow	<p>Nếu thông tin đăng ký không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng cung cấp lại thông tin</p> <ol style="list-style-type: none">4a.1. Hệ thống thông báo thông tin đăng ký không hợp lệ4a.2. Người dùng nhập lại thông tin đăng ký4a.3. Hệ thống kiểm tra thông tin đăng ký hợp lệ4a.4. Hệ thống xác nhận thông tin đăng ký hợp lệ và gửi thông báo đăng ký tài khoản thành công <p>Nếu người dùng đã có tài khoản trên trang web, hệ thống sẽ hiển thị thông báo và yêu cầu người dùng đăng ký bằng tài khoản khác</p> <ol style="list-style-type: none">4b.1. Hệ thống thông báo tài khoản đã tồn tại4b.2. Người dùng nhập lại tài khoản4b.3. Hệ thống kiểm tra thông tin đăng ký hợp lệ4b.4. Hệ thống xác nhận thông tin đăng ký hợp lệ và gửi thông báo đăng ký tài khoản thành công
Exception Flows	<p>Nếu hệ thống gặp lỗi kỹ thuật trong quá trình xử lý thông tin đăng ký, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng thử lại sau</p> <ol style="list-style-type: none">4c.1. Hệ thống hiện lỗi khi xử lý thông tin đăng ký4c.2. Hệ thống thông báo yêu cầu người dùng thử lại sau

4.2 Chính sửa thông tin người dùng

4.2.1 Use case diagram



Hình 2: Use case diagram cho tính năng chỉnh sửa thông tin người dùng

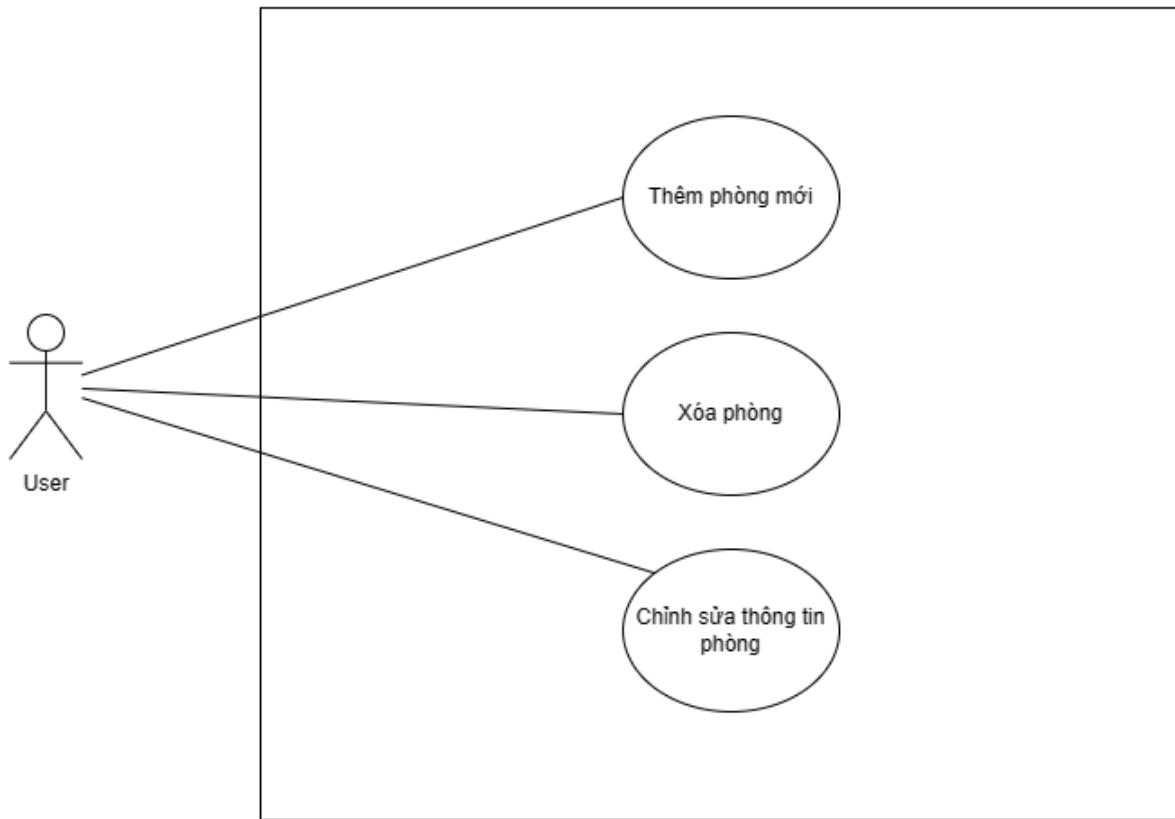


4.2.2 Use case scenario

Use case name	Chỉnh sửa thông tin tài khoản
Actor	Người dùng
Description	Tính năng cho phép người dùng chỉnh sửa thông tin cá nhân trên tài khoản của mình.
Preconditions	Người dùng đã đăng nhập vào tài khoản của mình.
Postconditions	Thông tin cá nhân của người dùng đã được cập nhật thành công trên tài khoản của mình
Trigger	Người dùng chọn mục thông tin cá nhân trên trang chính của giao diện
Main Flow	<ol style="list-style-type: none">Người dùng chọn chức năng "Chỉnh sửa thông tin cá nhân".Người dùng thực hiện việc chỉnh sửa thông tin cá nhân như tên, địa chỉ email, mật khẩu, số điện thoại,...Người dùng nhấn nút "Cập nhật" để cập nhật thông tin cá nhân.Hệ thống kiểm tra thông tin cá nhân và cập nhật nó trong cơ sở dữ liệu. Hệ thống hiển thị thông báo cập nhật thành công.
Alternative Flow	<p>Nếu người dùng không muốn thực hiện chỉnh sửa, họ có thể thoát ra khỏi giao diện chỉnh sửa thông tin người dùng</p> <p>2a.1. Người dùng nhấn nút Dashboard trên thanh điều hướng bên tay trái</p> <p>2a.2. Hệ thống chuyển hướng người dùng về trang chính</p> <p>Nếu thông tin cập nhật không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng cung cấp lại thông tin</p> <p>4a.1. Hệ thống thông báo thông tin chỉnh sửa không hợp lệ</p> <p>4a.2. Người dùng nhập lại thông tin</p> <p>4a.3. Hệ thống kiểm tra tính hợp lệ</p> <p>4a.4. Hệ thống xác nhận thông tin chỉnh sửa hợp lệ và gửi thông báo đã cập nhật thành công</p>
Exception Flows	<p>Nếu hệ thống gặp lỗi kỹ thuật trong quá trình cập nhật thông tin cá nhân, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng thử lại sau</p> <p>4b.1. Hệ thống hiện lỗi kỹ thuật trong quá trình cập nhật thông tin</p> <p>4b.2. Hệ thống thông báo yêu cầu người dùng thử lại sau</p>

4.3 Quản lý các phòng

4.3.1 Use case diagram



Hình 3: Use case diagram cho chức năng Quản lý các phòng

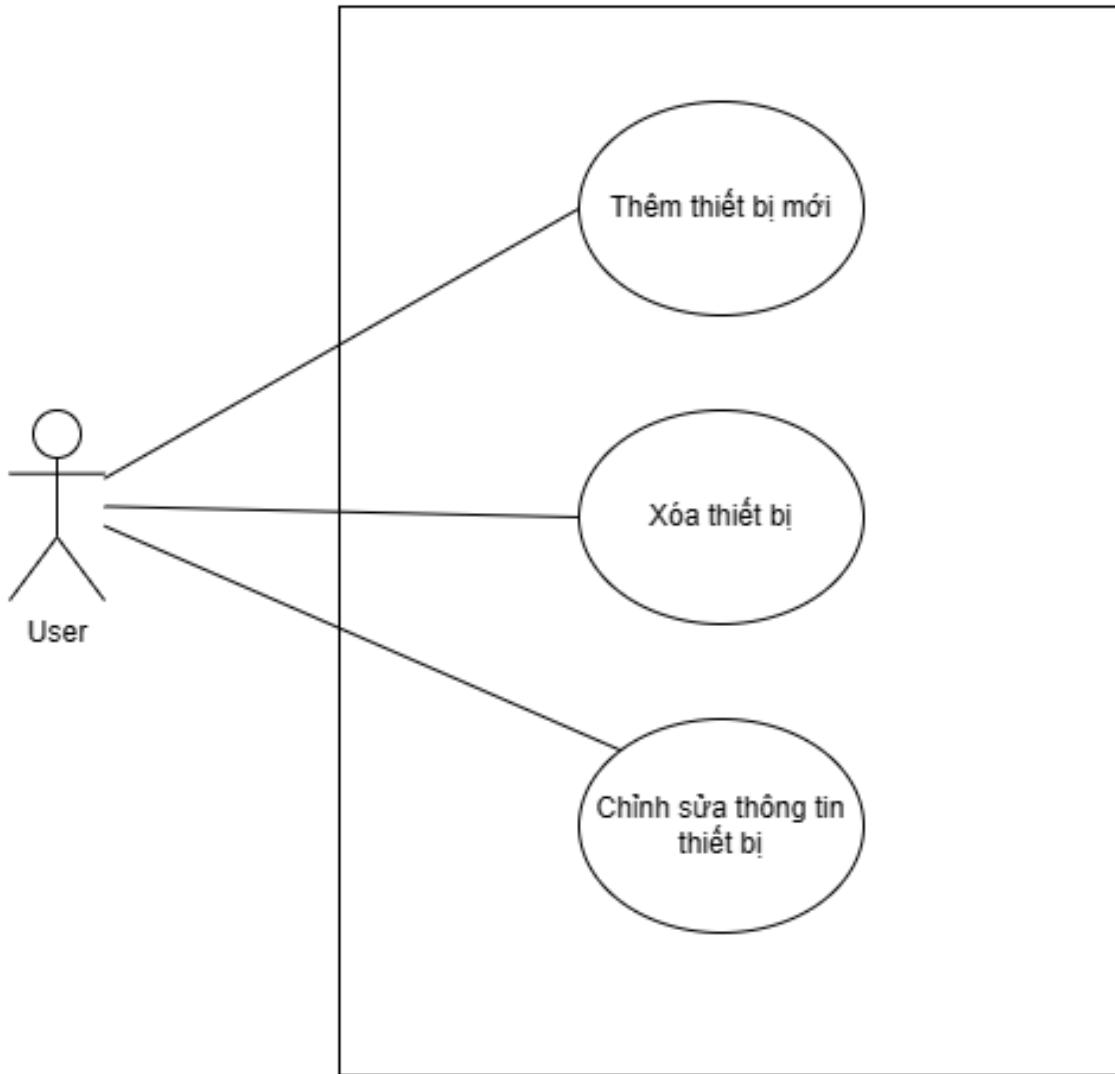


4.3.2 Use case scenario

Use case name	Quản lý phòng
Actor	Người dùng
Descriptions	Người dùng có thể thêm, xóa phòng, chỉnh sửa thông tin phòng
Triggers	Người dùng chọn mục Phòng
Preconditions	Người dùng đăng nhập thành công vào hệ thống
Postcondition	Một hay nhiều phòng được thêm, xóa hoặc chỉnh sửa thông tin tần suất tưới
Normal Flows	<ol style="list-style-type: none">Hệ thống hiển thị danh sách phòng hiện có lên giao diệnNgười dùng chọn thêm phòng mớiHệ thống hiển thị form đăng ký thông tin phòng mớiNgười dùng điền đầy đủ thông tin vào các ô nhập vàoNgười dùng chọn Xác nhận để hệ thống cập nhật ngôi nhàHệ thống quay trở lại trang danh sách phòng và hiển thị danh sách phòng được cập nhật
Alternative Flows	<p>Tại bước 1: Người dùng chọn chỉnh sửa thông tin phòng</p> <ol style="list-style-type: none">1a1. Hệ thống hiển thị các thông tin có sẵn của phòng1a2. Người dùng cập nhật thông tin cho phòng1a3. Người dùng chọn Xác nhận để hệ thống cập nhật thông tin phòng1a4. Quay trở lại bước 6 <p>Tại bước 1: Người dùng chọn xóa phòng</p> <ol style="list-style-type: none">1b1. Hệ thống nhận được thông tin và tiến hành xóa phòng đã chọn1b2. Quay trở lại bước 6
Exception Flows	<p>Tại bước 1: Hệ thống không hiển thị danh sách phòng</p> <ol style="list-style-type: none">1c1. Hệ thống hiển thị thông báo lỗi <p>Tại bước 6: Hệ thống gặp sự cố và không cập nhật thông tin phòng mới lên hệ thống được</p> <ol style="list-style-type: none">6a1. Hệ thống hiển thị thông báo lỗi <p>Tại bước 1a1: Hệ thống gặp sự cố và không hiển thị thông tin phòng có sẵn</p> <ol style="list-style-type: none">1a1.1: Hệ thống hiển thị thông báo lỗi

4.4 Quản lý thông tin thiết bị

4.4.1 Use case diagram



Hình 4: Use case diagram cho chức năng Quản lý thiết bị

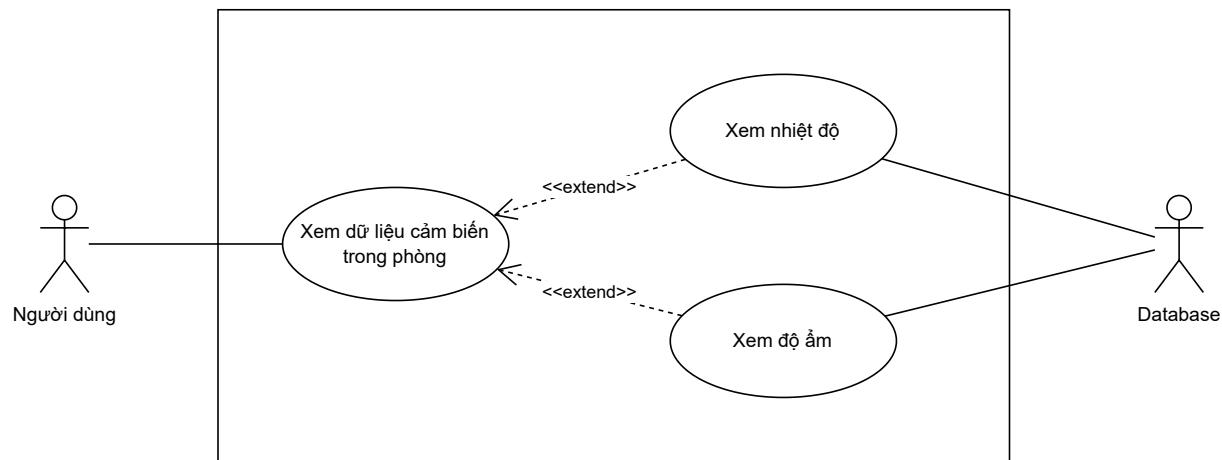
4.4.2 Use case scenario



<i>Use case name</i>	Quản lý thiết bị trong phòng
Actor	Người dùng
Descriptions	Người dùng có thể thêm, xóa các thiết bị trong phòng, chỉnh sửa thông tin hiển thị của thiết bị trên ứng dụng
Triggers	Người dùng chọn phòng mà mình quản lý
Preconditions	Người dùng đăng nhập thành công vào hệ thống
Postcondition	Một hay nhiều thiết bị trong phòng được thêm, xóa hoặc chỉnh sửa thông tin tần suất tươi
Normal Flows	<ol style="list-style-type: none">Người dùng chọn thêm thiết bị mớiHệ thống hiển thị form đăng ký thông tin thiết bị mớiNgười dùng điền đầy đủ thông tin vào các ô nhập vàoNgười dùng chọn Lưu để hệ thống cập nhật thiết bịHệ thống quay trở lại trang danh sách thiết bị và hiển thị thêm thiết bị mới
Alternative Flows	<p>Tại bước 1: Người dùng bấm vào thiết bị để chỉnh sửa thông tin thiết bị đó</p> <ol style="list-style-type: none">1a1. Hệ thống hiển thị bảng thông tin chi tiết của thiết bị đó1a2. Người dùng cập nhật thông tin cho thiết bị1a3. Người dùng chọn Lưu thay đổi để hệ thống cập nhật thông tin thiết bị1a4. Quay trở lại bước 5 <p>Tại bước 1: Người dùng bấm vào thiết bị để chọn xóa thiết bị đó khỏi ngôi nhà</p> <ol style="list-style-type: none">1b1. Hệ thống hiển thị bảng thông tin chi tiết của thiết bị đó1b2. Người dùng chọn xóa thiết bị1b3. Hệ thống sẽ hỏi lại người dùng có thật sự muốn xóa thiết bị này không1b4. Người dùng chọn Có để hệ thống xóa thiết bị1b5. Quay trở lại bước 5
Exception Flows	<p>Tại bước 5: Hệ thống gặp sự cố và không cập nhật thông tin thiết bị mới lên được</p> <ol style="list-style-type: none">5a1. Hệ thống hiển thị thông báo lỗi, các thông tin đã nhập vào được giữ nguyên trên giao diện <p>Tại bước 1a1: Hệ thống gặp sự cố và không hiển thị thông tin thiết bị</p> <ol style="list-style-type: none">1a1.1: Hệ thống hiển thị thông báo lỗi <p>Tại bước 1b1: Hệ thống gặp sự cố và không hiển thị thông tin thiết bị</p> <ol style="list-style-type: none">1b1.1: Hệ thống hiển thị thông báo lỗi và yêu cầu người dùng chọn lại thiết bị

4.5 Hiển thị dữ liệu cảm biến, trạng thái thiết bị lên màn hình

4.5.1 Use case diagram



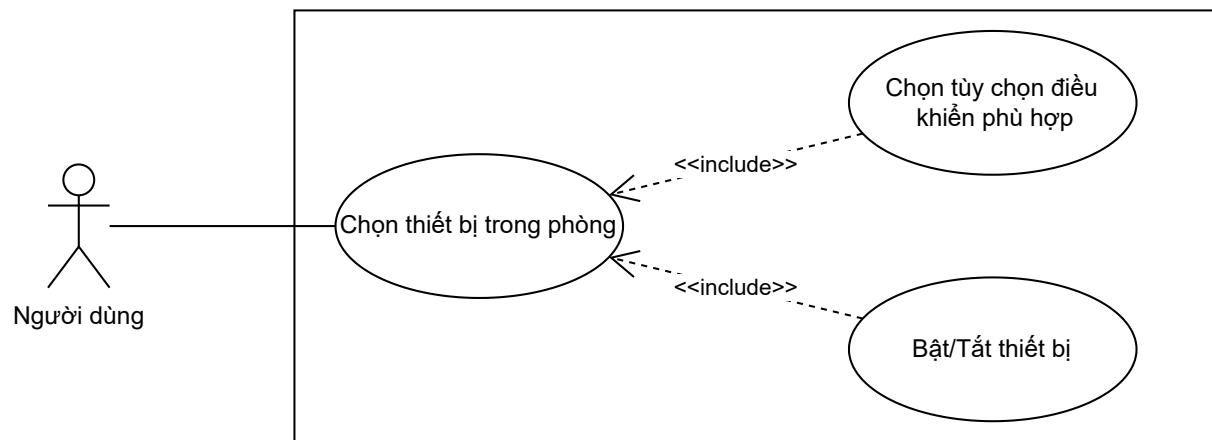
Hình 5: Use case diagram cho việc hiển thị dữ liệu cảm biến và trạng thái thiết bị lên màn hình

4.5.2 Use case scenario

Use case name	Hiển thị nhiệt độ và độ ẩm trong phòng
Actor	Người dùng
Description	Mô tả cách sử dụng hệ thống nhà thông minh để hiển thị nhiệt độ và độ ẩm trong phòng.
Triggers	Người dùng chọn mục Trạng thái nhà
Preconditions	Người dùng đăng nhập thành công vào hệ thống và nhiệt độ, độ ẩm đã được đo và đã gửi về hệ thống
Postconditions	Người dùng đã xem được nhiệt độ và độ ẩm hiện tại của phòng
Normal Flow	<ol style="list-style-type: none"> 1. Người dùng chọn phòng cần xem nhiệt độ và độ ẩm 2. Hệ thống hiển thị nhiệt độ và độ ẩm hiện tại của phòng trên giao diện người dùng
Alternative Flows	
Exception Flows	<p>Tại bước 2: Hệ thống gặp sự cố và không thể lấy dữ liệu từ database</p> <p>2a: Hệ thống hiển thị thông báo lỗi</p>

4.6 Điều khiển thiết bị trong nhà

4.6.1 Use case diagram



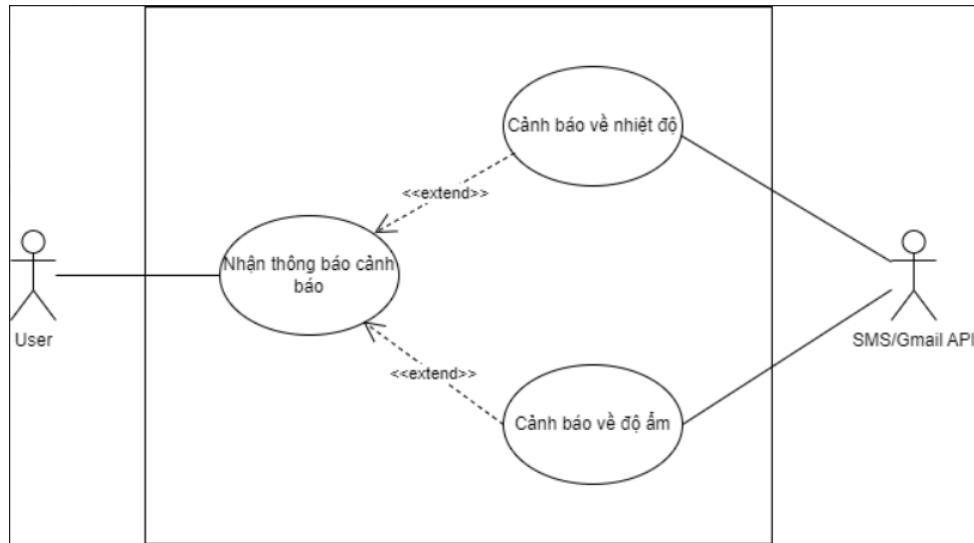
Hình 6: Use case diagram cho điều khiển thiết bị trong nhà

4.6.2 Use case scenario

Use case name	Điều khiển các thiết bị trong nhà
Actor	Người dùng
Description	Mô tả cách sử dụng hệ thống nhà thông minh để điều khiển các thiết bị trong nhà
Triggers	Người dùng chọn mục Thiết bị
Preconditions	Người dùng đăng nhập thành công vào hệ thống và các thiết bị đã được kết nối với hệ thống
Postconditions	Các thiết bị trong nhà đã được điều khiển theo yêu cầu của người dùng
Normal Flow	Người dùng chọn phòng cần điều khiển thiết bị Hệ thống hiển thị danh sách các thiết bị trong phòng trên giao diện người dùng Người dùng chọn thiết bị mà mình muốn điều khiển Hệ thống hiển thị các tùy chọn điều khiển cho thiết bị đó, bao gồm các nút bật/tắt, điều chỉnh độ sáng, nhiệt độ,... Người dùng lựa chọn tùy chọn điều khiển phù hợp Hệ thống gửi thông tin đến thiết bị và quay lại giao diện danh sách các thiết bị trong phòng
Alternative Flows	
Exception Flows	Tại bước 5: Điều khiển thiết bị không thành công do thiết bị hỏng hoặc không kết nối đến hệ thống 5a. Hệ thống hiển thị thông báo lỗi, yêu cầu người dùng thực hiện lại bước 3 hoặc kiểm tra thiết bị 5b. Người dùng thực hiện lại bước 3 hoặc kiểm tra lại thiết bị

4.7 Thông báo khi nhiệt độ, độ ẩm vượt ngưỡng quy định

4.7.1 Use case diagram



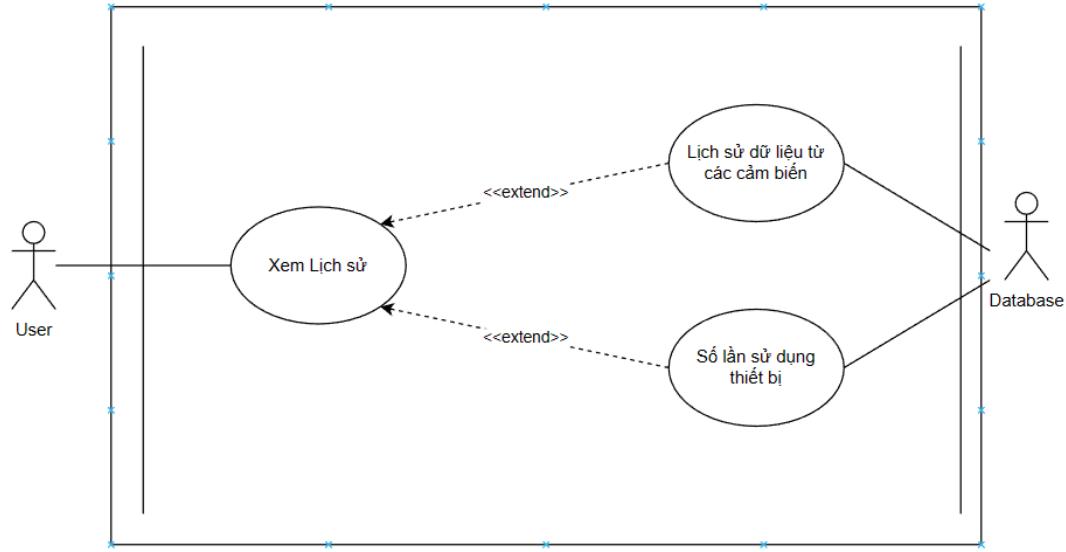
Hình 7: Use case diagram Thông báo khi nhiệt độ, độ ẩm vượt ngưỡng quy định

4.7.2 Usecase scenario

Use case name	Warning temp and humi.
Actor	User
Description	Thông báo khi nhiệt độ và độ ẩm vượt ngưỡng quy định
Triggers	Nhiệt độ và độ ẩm vượt ngưỡng quy định
Preconditions	User đang ở giao diện Màn hình chính
Postconditions	Gửi thông báo tới User
Normal Flow	User đang ở giao diện Màn hình chính. Gửi thông báo cho User khi nhiệt độ và độ ẩm vượt ngưỡng quy định. User chọn Đóng để tắt thông báo task.
Alternative Flows	Không
Exception Flows	Tại bước 2: Hệ thống gặp sự cố và không thể lấy dữ liệu từ database 2a. Hệ thống hiển thị thông báo lỗi. Tại bước 2: Dữ liệu nhiệt độ và độ ẩm chưa có trong hệ thống 2a: Hệ thống hiển thị thông báo lỗi, yêu cầu kiểm tra lại cảm biến.

4.8 Xem lịch sử ghi nhận sử dụng thiết bị và cảm biến

4.8.1 Use case diagram



Hình 8: Use case diagram cho tính năng Xem lịch sử ghi nhận sử dụng thiết bị và cảm biến

4.8.2 Usecase scenario



Use case name	Xem lịch sử ghi nhận sử dụng thiết bị và cảm biến
Actor	Người dùng
Description	Người dùng xem lại lịch sử dữ liệu đo được của các cảm biến và lịch sử ghi nhận sử dụng thiết bị
Triggers	Người dùng chọn mục Lịch sử
Preconditions	Người dùng đăng nhập thành công vào hệ thống
Postconditions	Người dùng đã xem được lịch sử dữ liệu của các cảm biến đo được và lịch sử ghi nhận sử dụng thiết bị
Normal Flow	<ol style="list-style-type: none">Người dùng chọn phòng, sau đó hệ thống sẽ cập nhật danh sách thiết bị và cảm biến nằm trong phòng đóNgười dùng chọn thiết bị, có thể tùy chọn ngày bắt đầu và ngày kết thúc để giới hạn tập dữ liệuNgười dùng chọn Cập nhật để hệ thống hiển thị đồ thị dữ liệu và bảng dữ liệu
Alternative Flows	<p>Tại bước 1: Người dùng chưa chọn phòng nhưng chọn Cập nhật</p> <ol style="list-style-type: none">Hệ thống sẽ không cập nhật dữ liệu mới lên giao diện <p>Tại bước 2: Người dùng chưa chọn thiết bị nhưng chọn Cập nhật</p> <ol style="list-style-type: none">Hệ thống sẽ không cập nhật dữ liệu mới lên giao diện <p>Tại bước 3: Người dùng chọn nút tải về</p> <ol style="list-style-type: none">Hệ thống tự động tạo một file có đuôi csv với đầu vào là tập dữ liệu đang có trên giao diện, sau đó được tải về máy của người dùng
Exception Flows	<p>Tại bước 3: Hệ thống không truy xuất được database để cập nhật tập dữ liệu</p> <ol style="list-style-type: none">Hệ thống đưa ra thông báo lỗi và yêu cầu người dùng tải lại trang

Phần II

Implementation

5 Phần cứng

5.1 Các thiết bị sử dụng

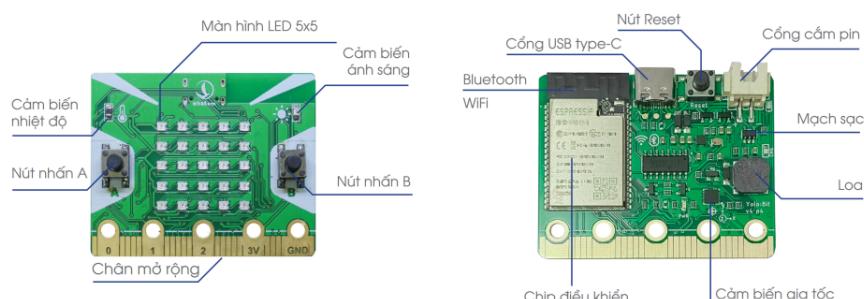
5.1.1 Mạch lập trình

Mạch Yolo:Bit

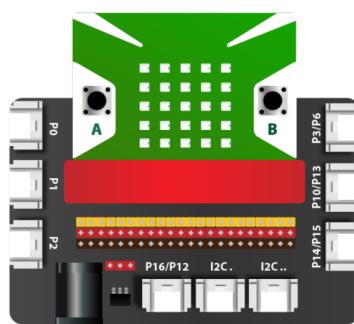
Yolo:Bit là một máy tính nhúng có hệ điều hành nhỏ (tên là Micro Python), sử dụng điện áp thấp 3.3V với các góc cạnh được bo tròn được chạy trên nền tảng chip ESP32. Bên cạnh nhiều ngoại vi và cảm biến tích hợp, Yolo:Bit còn tích hợp sẵn khả năng kết nối không dây (WiFi và Bluetooth), nhằm hỗ trợ cho các ứng dụng kết nối vạn vật - Internet of Things - một công nghệ rất phổ biến trong các ứng dụng ngày nay.

Mạch mở rộng

Sử dụng kết hợp với mạch Yolo:Bit để có thể kết nối với nhiều thiết bị ngoại vi



Hình 9: Thiết kế Yolo:Bit



Hình 10: Yolo:Bit và mạch mở rộng

5.1.2 Các thiết bị ngoại vi

Các thiết bị đầu vào(input)

- Mắt nhận hồng ngoại

Sử dụng kết hợp với remote điều khiển, mắt nhận hồng ngoại sẽ nhận sóng hồng ngoại từ remote và cho ra tín hiệu tương ứng.



Hình 11: Mắt nhận hồng ngoại

- Cảm biến nhiệt độ độ ẩm DHT20

DHT20 là một thiết bị cảm biến rất phổ biến đối với các ứng dụng dân dụng để đo giá trị nhiệt độ, độ ẩm; với ưu điểm là giá thành thấp và cách sử dụng đơn giản. Dãy đo của cảm biến khá phù hợp trong môi trường bình thường không có biến động lớn, với độ ẩm trong khoảng 20 - 90 % và nhiệt độ là 0 - 50°C.



Hình 12: Cảm biến nhiệt độ, độ ẩm

- Cảm biến ánh sáng

Hoạt động dựa trên nguyên lý trở kháng để phản ánh thông số cần đo đặc, cụ thể hơn là quang trở (thiết bị có điện trở thay đổi khi cường độ ánh sáng thay đổi).



Hình 13: Cảm biến ánh sáng

Các thiết bị đầu ra (output)

- **Dèn 4 LED RGB**

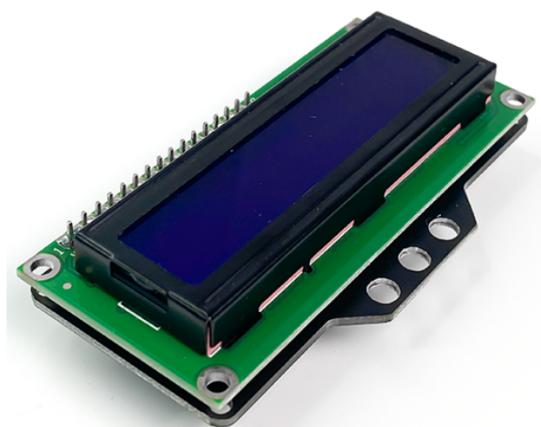
Module 4 LED RGB là 1 dãy 4 đèn nối tiếp nhau, gọi là NeoPixel. Toàn bộ dây LED có thể mở rộng lên đến 144 đèn, mỗi đèn có thể điều khiển được màu sắc riêng của nó từ 3 màu cơ bản RGB.



Hình 14: 4 LED RGB

- **Màn hình LCD 16x2**

Thiết bị mà chúng ta sử dụng có thể hiển thị 2 dòng với mỗi dòng tối đa 16 ký tự, nên còn được gọi là màn hình LCD 16 x 2. Tuy nhiên, để điều khiển được thiết bị này, cần phải sử dụng rất nhiều chân kết nối (tối thiểu 3 chân điều khiển và 4 chân tín hiệu). Do vậy, để tối ưu trong việc thiết kế phần cứng, một mạch hỗ trợ có tên là I2C đã được thiết kế đi kèm với màn hình LCD. Do I2C là một dạng giao tiếp đặc biệt, nên chúng ta chỉ có thể cắm dây nối giữa màn hình LCD và mạch mở rộng tại chân cắm I2C.



Hình 15: Màn hình LCD 16x2

- Quạt mini

Thiết bị quạt mini này sử dụng động cơ điện một chiều, hay còn gọi là động cơ DC (Direct Current - Dòng điện một chiều). Không giống như các thiết bị chỉ có nhu cầu bật và tắt (chẳng hạn như bóng đèn), quạt là thiết bị cần phải điều khiển được tốc độ. Điều áp hay điều xung, là phương pháp cơ bản để điều khiển tốc độ của động cơ. Nhờ kỹ thuật này, điện thế tại một chân của Yolo:Bit có thể thay đổi được điện áp, từ đó thay đổi cường độ dòng điện qua động cơ và thay đổi tốc độ quay của động cơ.



Hình 16: Quạt mini

- Động cơ servo

RC Servo là một dạng động cơ được điều khiển theo góc. Hoàn toàn trái ngược với động cơ cánh quạt, RC Servo có tốc độ thấp nhưng có lực khá mạnh. Bên trong cấu tạo của RC Servo thường đi kèm với bộ hộp số giảm tốc có tỉ số truyền lớn. Do vậy, chỉ cần một lực nhỏ ở phía động cơ, có thể truyền động ra một lực lớn ở cơ cấu truyền động.

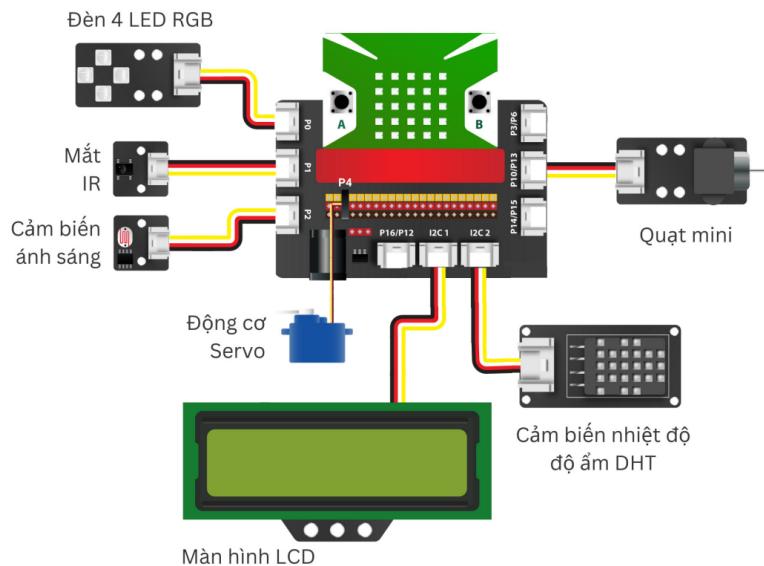


Hình 17: Động cơ Servo

5.1.3 Kết nối các thiết bị

	Chức năng	Thiết bị	Kết nối khẩn dĩ	Kết nối thực tế
Quan trắc môi trường	Nhiệt độ	DHT20	I2C1, I2C2	I2C2
	Độ ẩm			
	Ánh sáng	Cảm biến ánh sáng	P0, P1, P2	P2
	Hiển thị	LCD 16x2	I2C1, I2C2	I2C1
	Điều hòa nhiệt độ	Quạt mini	Tất cả	P10
Cửa mật mã	Nhập mật mã	Remote và mắt nhận	Tất cả	P1
	Đóng, mở cửa	Động cơ RC	Tất cả	P3
Cảnh báo	Đèn báo hiệu	Đèn 3 màu RGB	Tất cả	P0

Hình 18: Các thiết bị sử dụng



Hình 19: Kết nối phần cứng cho dự án

5.2 Các tính năng triển khai

5.2.1 Tổng quan tính năng

- Tính năng cửa mật mã (sử dụng với remote - mắt nhận hồng ngoại hoặc trực tiếp trên app)
- Giám sát chất lượng không khí - Đo và cập nhật thường xuyên các dữ liệu như nhiệt độ, độ ẩm và ánh sáng và thể hiện trực quan bằng bảng đồ
- Điều khiển thiết bị trên ứng dụng (đèn và quạt)
- Kiểm tra nhiệt độ và ánh sáng với ngưỡng được thiết lập và thực hiện 1 số tính năng tự động
- Hẹn giờ bật, tắt đèn

5.2.2 Các khối lệnh

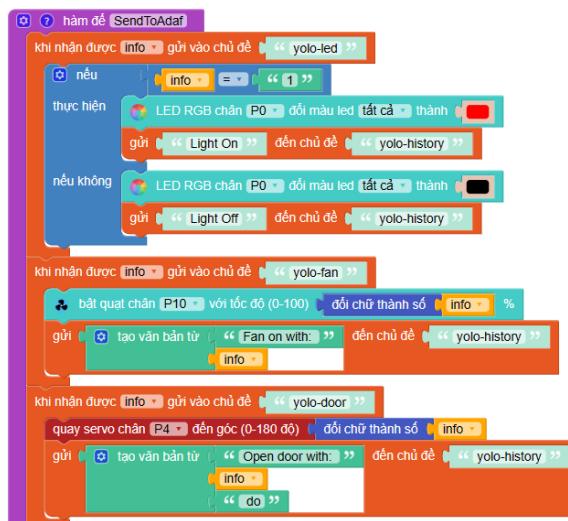
1. Khối lệnh bắt đầu



Hình 20: Khối lệnh 1

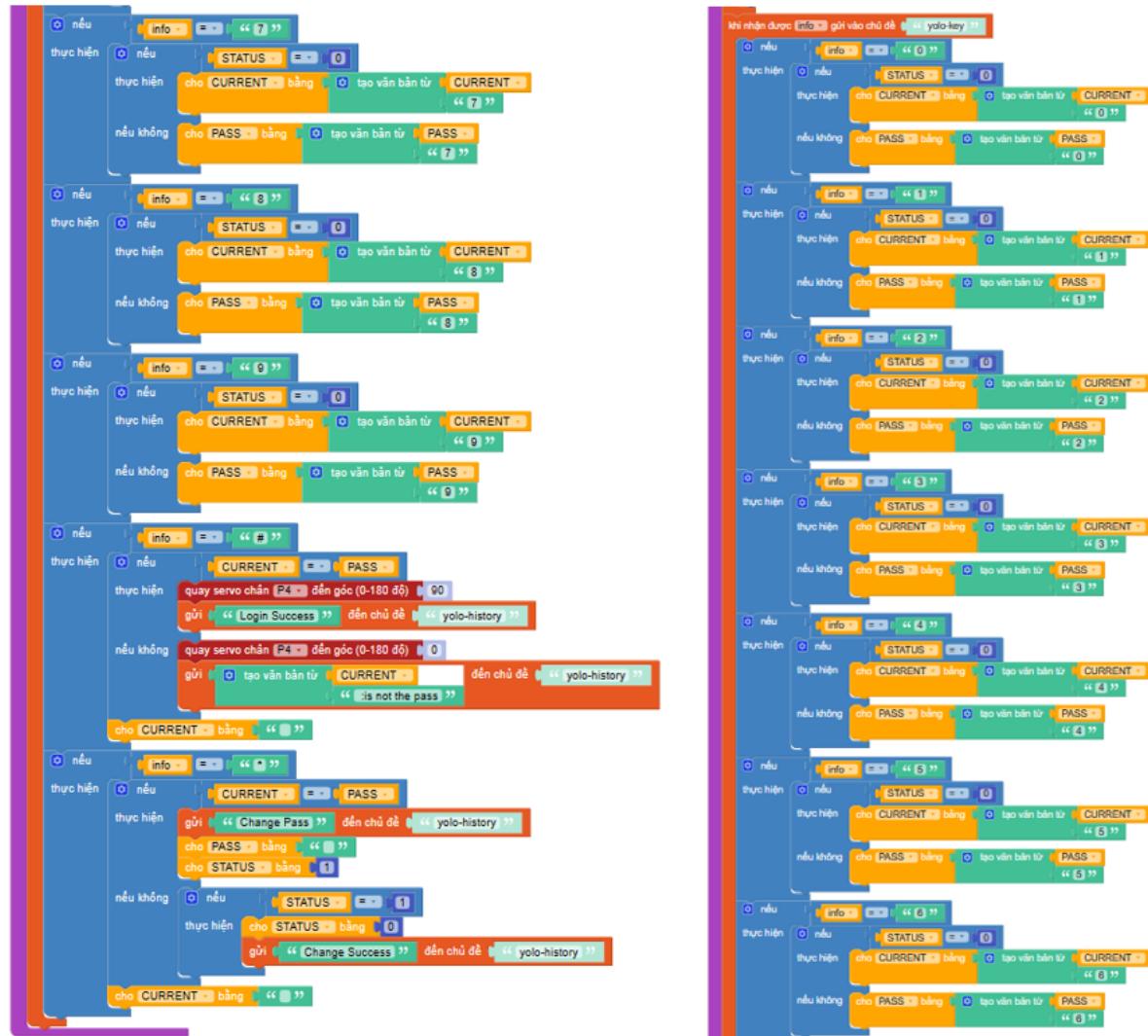
- Kết nối mạch với wifi và server Adafruit
- Cập nhật thời gian từ thư viện NTP, khởi tạo các biến sử dụng, và chạy hàm nhận thông tin từ kênh (SendToAdaf)
- Chạy các lệnh xử lý sự kiện và cập nhật thông tin từ Server trong vòng lặp

2. Khối lệnh quản lý các kênh dữ liệu

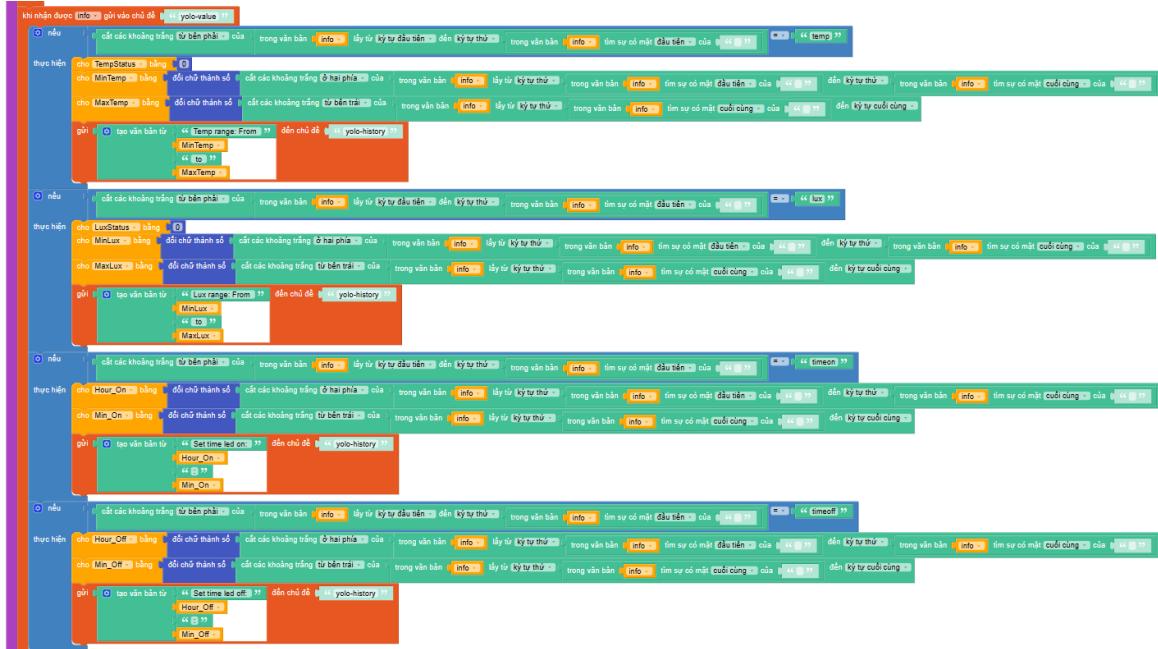


Hình 21: Khối lệnh 2.1

- Kênh yolo-led để thực hiện bật tắt đèn



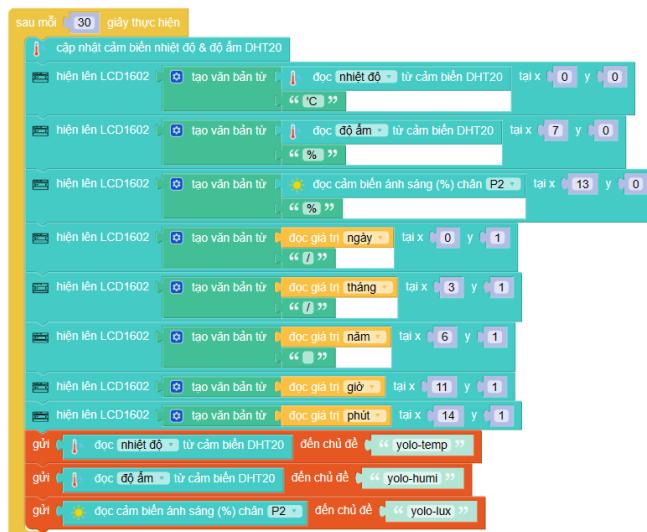
Hình 22: Khối lệnh 2.2



Hình 23: Khối lệnh 2.3

- Kênh yolo-fan để thực hiện bật tắt quạt
- Kênh yolo-door để xoay servo (cửa)
- Kênh yolo-vale để thay đổi giá trị ngưỡng của nhiệt độ, ánh sáng và hẹn giờ
- Kênh yolo-key quản lý việc đóng mở cửa bằng cửa mật mã

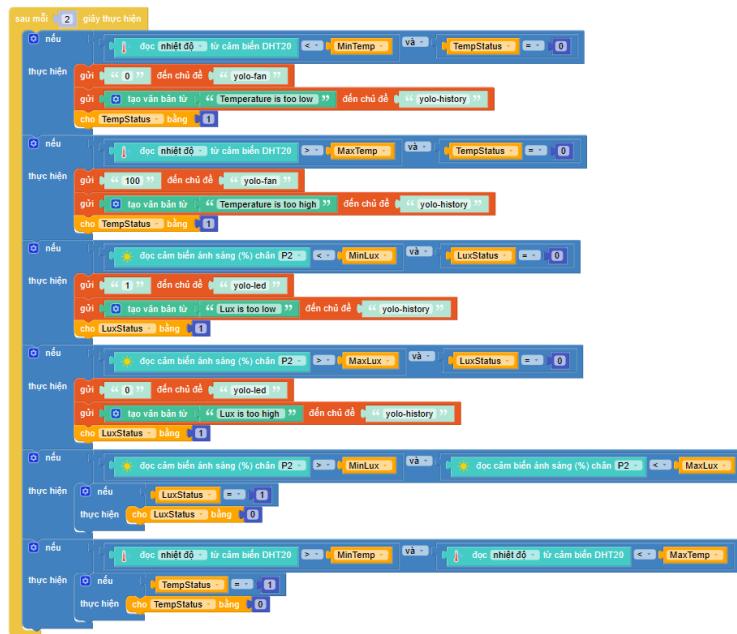
3. Khối lệnh cập nhật giá trị cảm biến DHT20, ánh sáng và hiển thị lên LCD



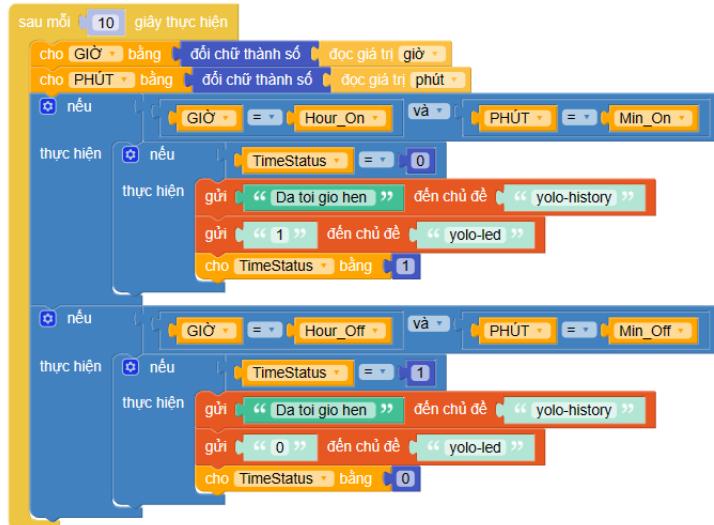
Hình 24: Khối lệnh 3

- Cập nhật dữ liệu nhiệt độ, độ ẩm và ánh sáng mỗi 30 giây và gửi lên Server
- Hiển thị nhiệt độ, độ ẩm, ánh sáng cùng với ngày giờ lên màn hình LCD

4. Khối lệnh quản lý giá trị ngưỡng và hẹn giờ



Hình 25: Khối lệnh 4.1



Hình 26: Khối lệnh 4.2

- Bật tắt đèn hoặc quạt nếu giá trị đo được không nằm trong ngưỡng thiết lập
- Hẹn giờ để bật, tắt đèn

5. Khối lệnh quản lý cửa mật mã sử dụng remote



Hình 27: Khối lệnh 5

5.3 Source Code

Link: <https://github.com/DanhNguyen02/SmartHome/tree/main/CodeHardware>



6 Phần mềm

6.1 Thiết kế cơ sở dữ liệu

Nhóm sẽ sử dụng MongoDB để thiết kế cơ sở dữ liệu sử dụng cho dự án lần này.

Cơ sở dữ liệu của dự án bao gồm các collection như sau:

- Các collection dùng để lưu trữ dữ liệu nhận được từ server Adafruit của nhóm: Nhóm sẽ lưu trữ dữ liệu từ 2 thiết bị bao gồm quạt, đèn và 2 cảm biến nhiệt độ và độ ẩm. Từng collection bao gồm 2 trường:

- _id: Đối tượng ObjectId mặc định của MongoDB
- time: Thời gian server hệ thống nhận được và xử lý dữ liệu để gửi lên database
- message: Nội dung dữ liệu được gửi về từ server Adafruit

Những collection đó có tên như sau:

- haiche198(feeds/yolo-fan)
- haiche198(feeds/yolo-humi)
- haiche198(feeds/yolo-led)
- haiche198(feeds/yolo-temp)

- user: Collection chính dùng để lưu trữ dữ liệu của hệ thống, bao gồm các trường sau:

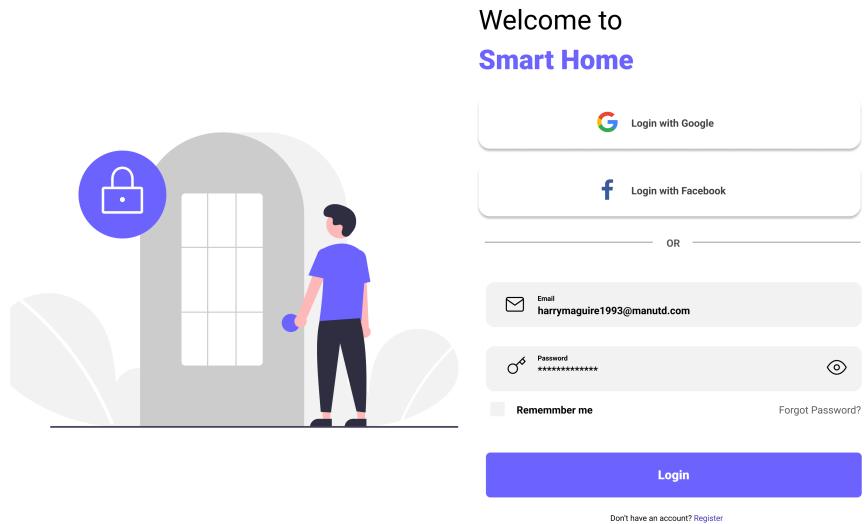
- _id: Đối tượng ObjectId mặc định của MongoDB
- name: Tên người dùng
- email: Địa chỉ email của người dùng, đồng thời cũng là tên tài khoản
- password: Một chuỗi ký tự đã được mã hóa từ mật khẩu thật của người dùng
- gender: Giới tính người dùng
- adafruitUsername: Tên tài khoản Adafruit mà người dùng sử dụng trong hệ thống
- adafruitActivekey: Active key của Adafruit mà người dùng sử dụng để truy cập vào các feed trong đó
- dob: Ngày sinh người dùng
- phone: Số điện thoại người dùng
- address: Địa chỉ người dùng
- avatar: Ảnh đại diện của người dùng
- timestamp: Thời gian tạo tài khoản



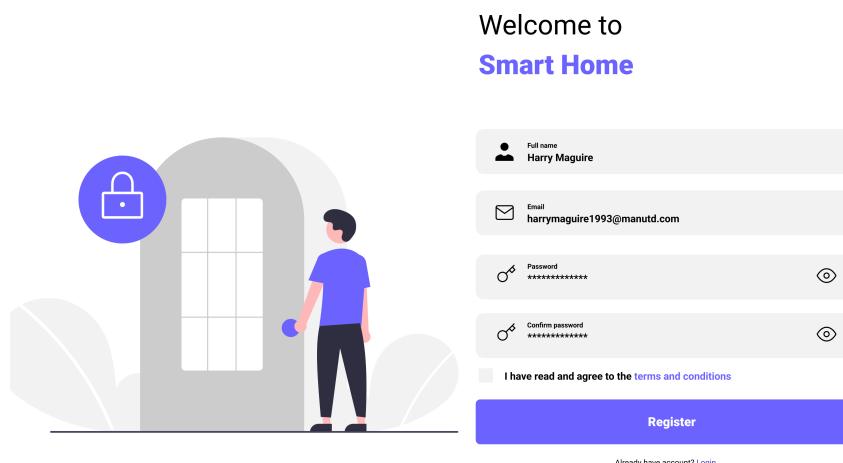
- rooms: Danh sách các object chứa thông tin các phòng ảo được người dùng thiết lập, trong đó bao gồm các thông tin sau:
 - name: Tên phòng do người dùng thiết lập
 - desc: Mô tả phòng do người dùng thiết lập
 - devices: Danh sách các Object về thiết bị và cảm biến được người dùng thiếp lập vào căn phòng tương ứng. Dữ liệu trong trường này bao gồm các thông tin sau:
 - * name: Tên của thiết bị
 - * feed: Link feed ứng với thiết bị. Đây là trường giúp nhận biết thiết bị sẽ nhận dữ liệu từ feed nào trên Adafruit
 - * type: Loại thiết bị
 - * min: Ngưỡng dưới của thông số thiết bị do người dùng thiết lập
 - * max: Ngưỡng trên của thông số thiết bị do người dùng thiết lập
- noti: Dữ liệu về các thông báo khi thông số của cảm biến vượt ngưỡng cho phép do người dùng cài đặt, bao gồm các thông tin sau:
 - time: Thời gian server hệ thống nhận được và xử lý dữ liệu để gửi lên database
 - room: Tên phòng chứa thiết bị nhận được thông báo
 - device: Tên thiết bị nhận được thông báo
 - type: Loại thiết bị
 - data: Dữ liệu nhận được khi đã xét ra khỏi ngưỡng quy định

6.2 Thiết kế UI

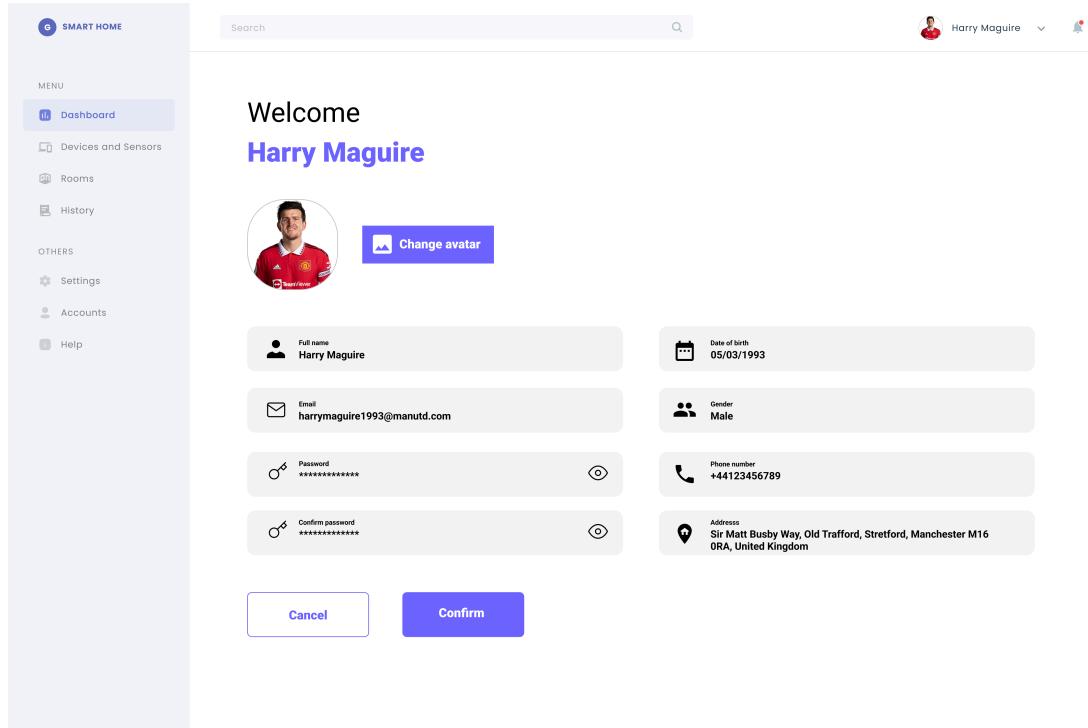
6.2.1 Trang đăng nhập, đăng ký và chỉnh sửa thông tin người dùng



Hình 28: Giao diện đăng nhập hệ thống

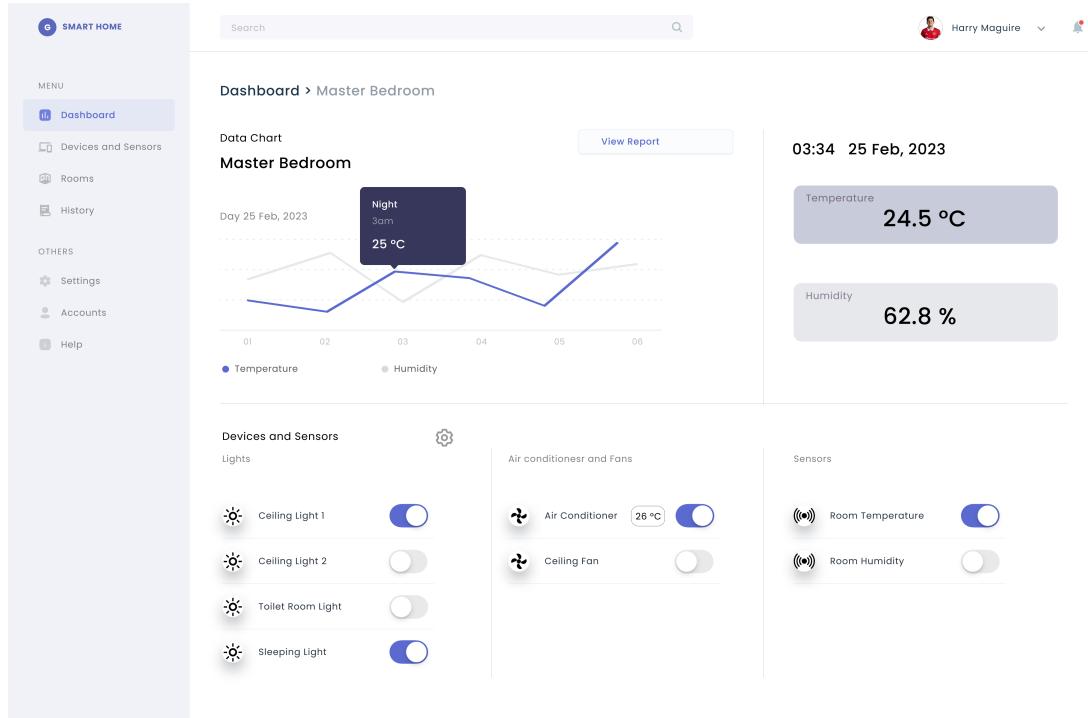


Hình 29: Giao diện đăng ký tài khoản



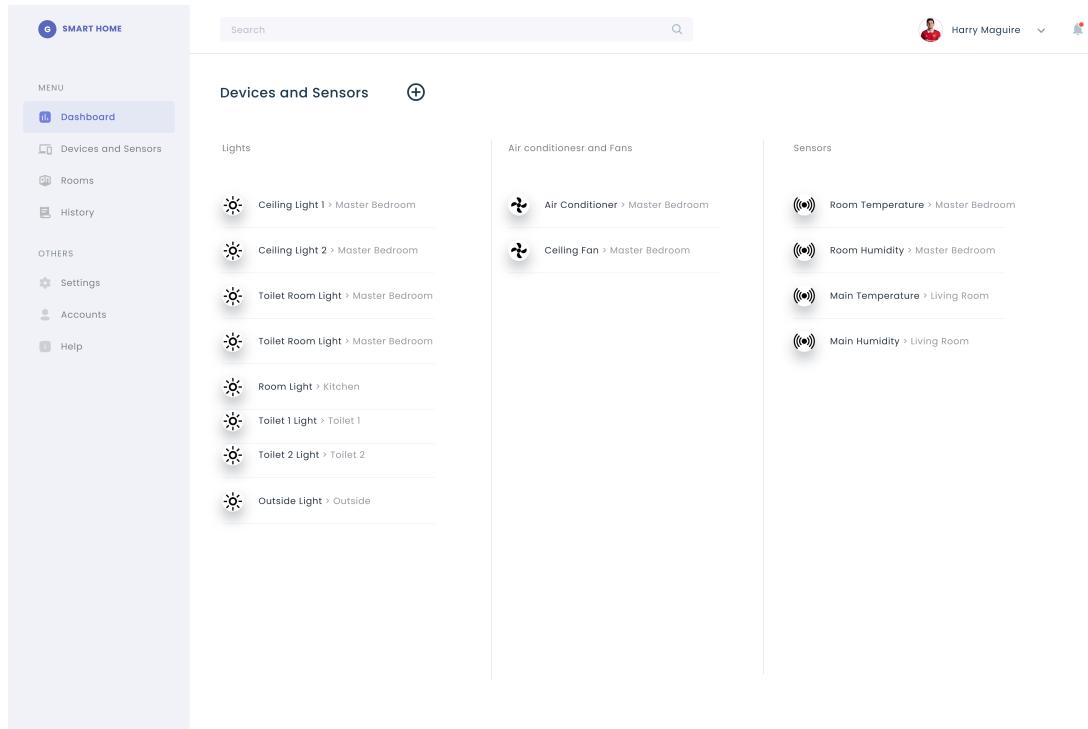
Hình 30: Giao diện chỉnh sửa thông tin tài khoản

6.2.2 Trang dashboard

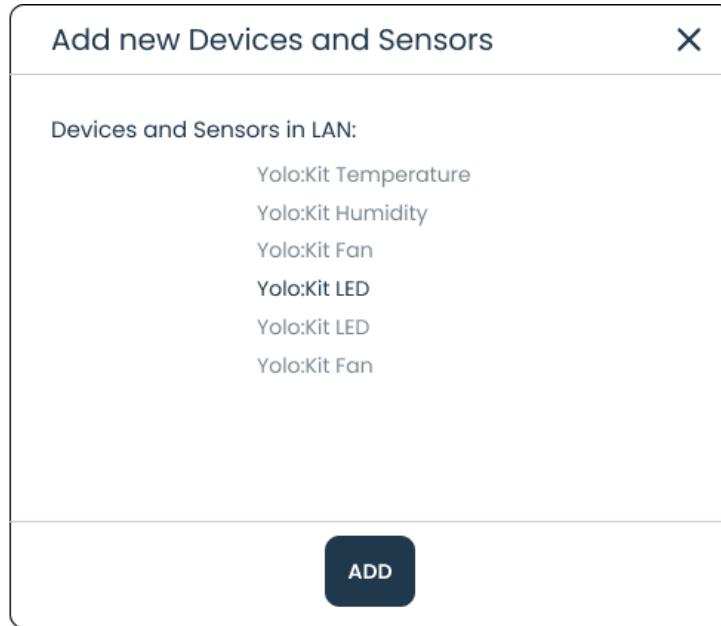


Hình 31: Giao diện dashboard

6.2.3 Trang thiết bị và cảm biến



Hình 32: Giao diện danh sách thiết bị và cảm biến



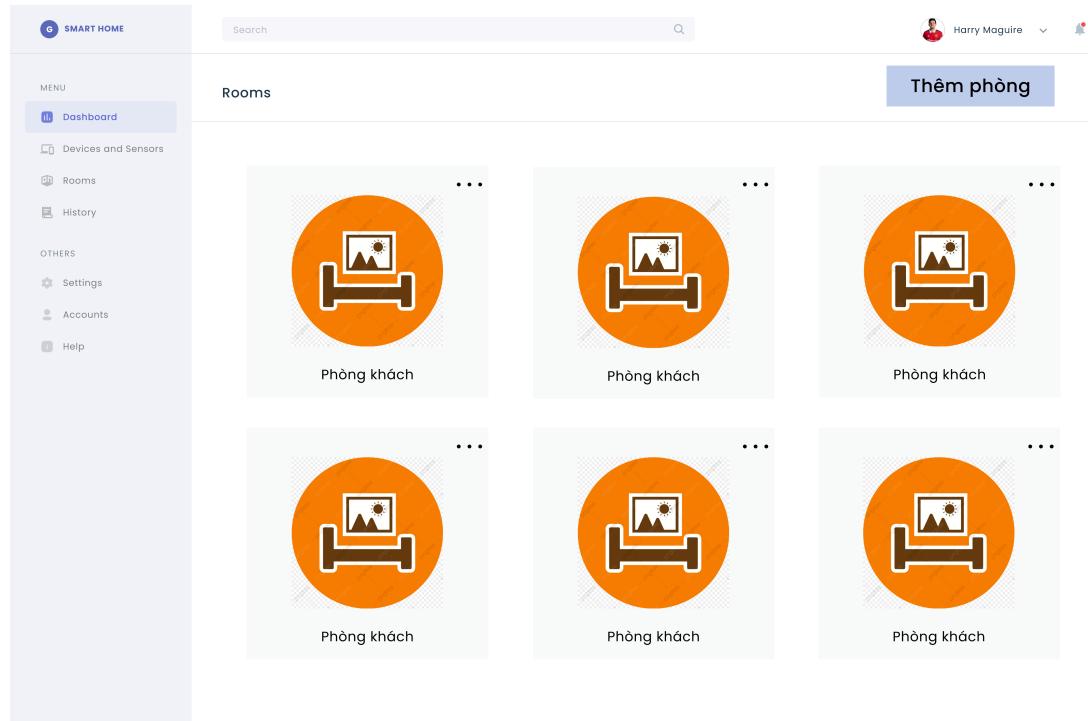
Hình 33: Giao diện thêm thiết bị và cảm biến

The image displays six wireframe forms arranged in two rows of three. Each form has a close button ('X') in the top right corner.

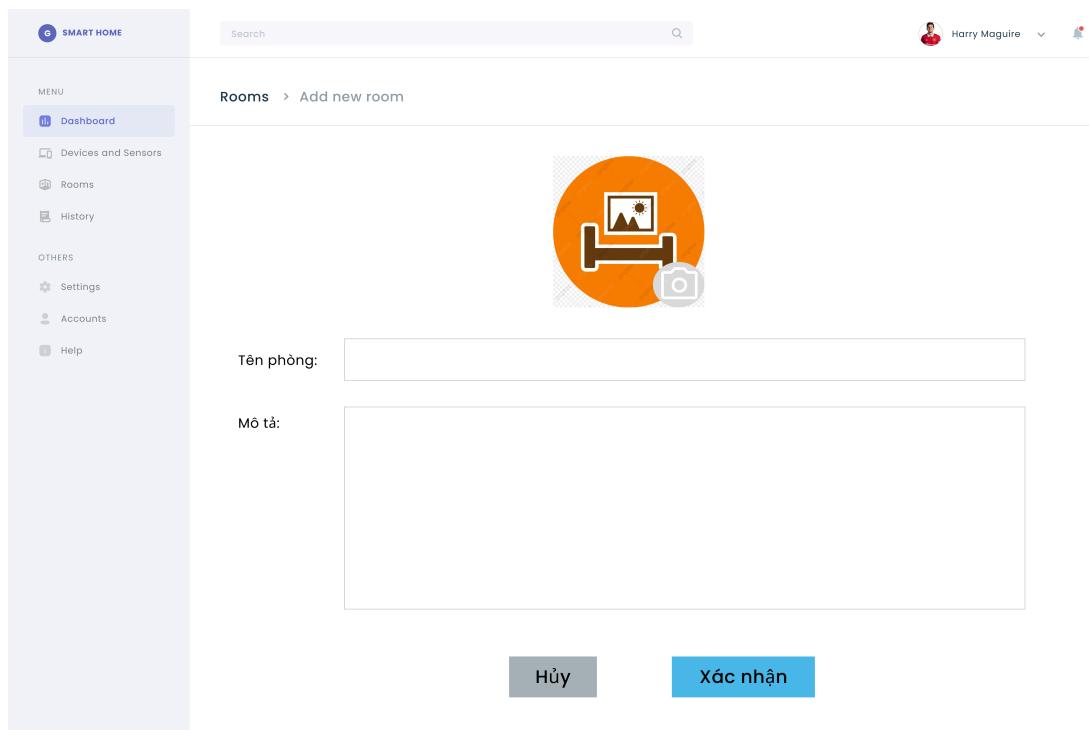
- Ceiling Light 1 > Master Bedroom**: Fields include Tên (Ceiling Light 1), Mô tả (Mô tả), Phòng (Master Bedroom), Turn on in (Auto, 18:00), Turn off in (5:30), and a 'cập nhật' button.
- Room Temperature > Master Bedroom**: Fields include Tên (Room Temperature), Mô tả (Mô tả), Phòng (Master Bedroom), Temperature (Temperature, 36), Max value (36), Min Value (21), and a 'cập nhật' button.
- Air Conditioner > Master Bedroom**: Fields include Tên (Air Conditioner), Mô tả (Mô tả), Phòng (Master Bedroom), Air Conditioner (Fan), and a 'cập nhật' button.
- Ceiling Light 1 > Master Bedroom**: Similar to the first form, but 'Turn on in' dropdown shows 'Manual' instead of 'Auto'.
- Room Temperature > Master Bedroom**: Fields include Tên (Room Temperature), Mô tả (Mô tả), Phòng (Master Bedroom), Temperature (Humidity), Max value (36), Min Value (21), and a 'cập nhật' button.
- Air Conditioner > Master Bedroom**: Fields include Tên (Air Conditioner), Mô tả (Mô tả), Phòng (Master Bedroom), Air Conditioner (Fan), and a 'cập nhật' button.

Hình 34: Các mục thêm thiết bị và cảm biến

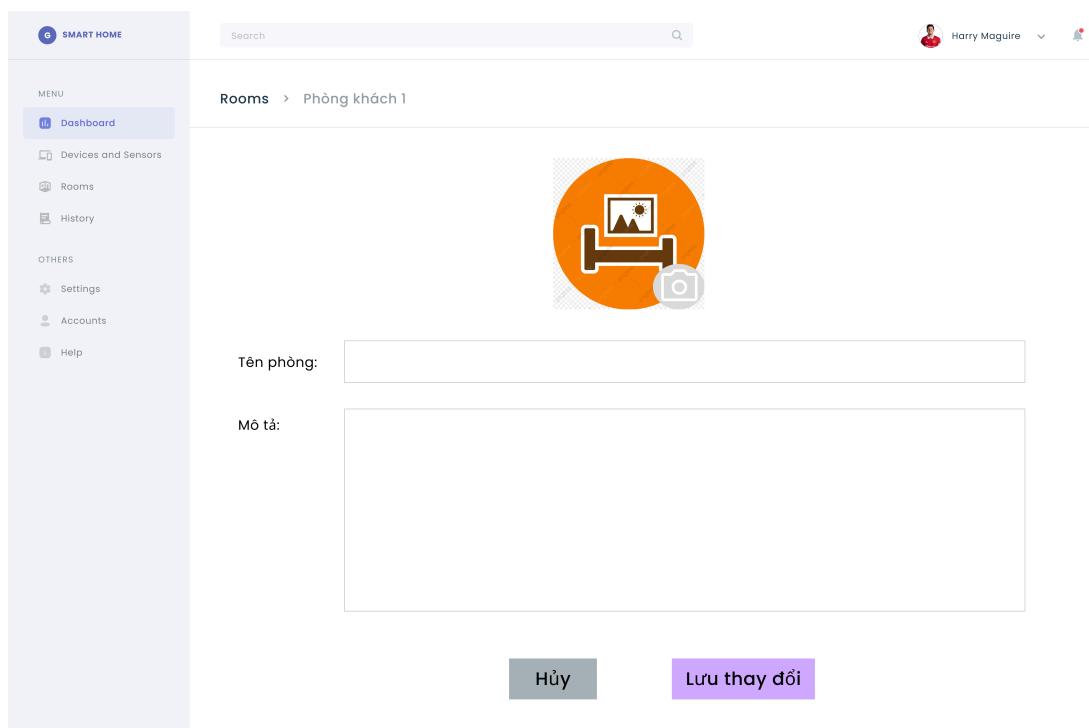
6.2.4 Giao diện xem thông tin phòng



Hình 35: Giao diện chọn phòng

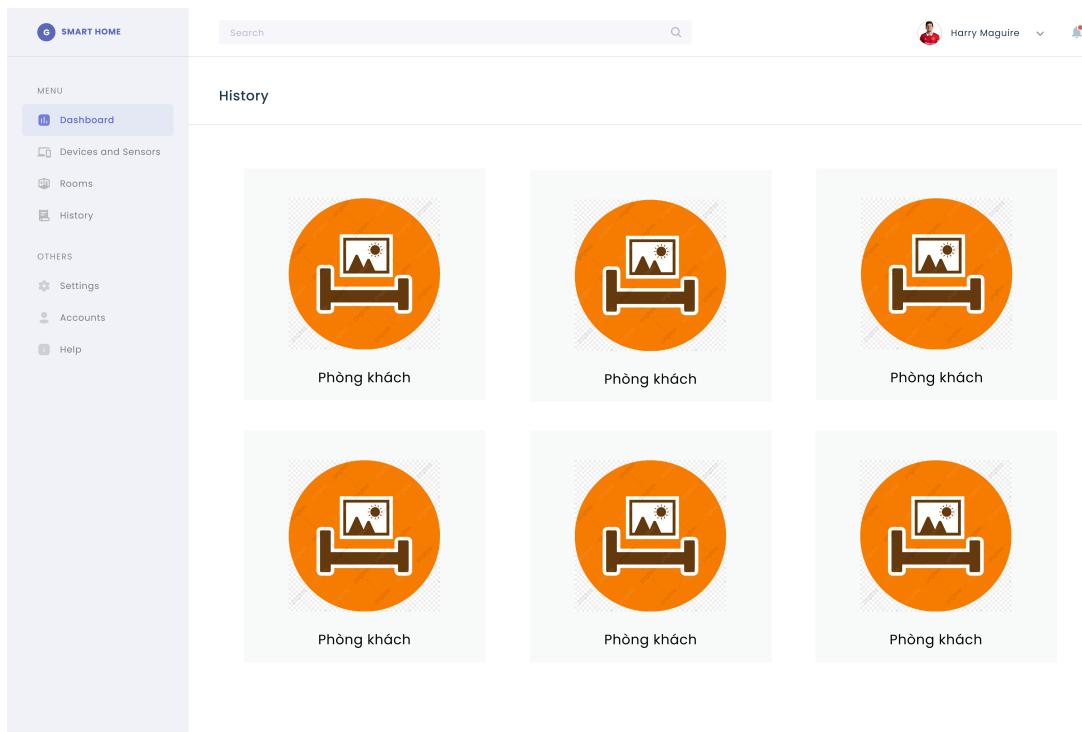


Hình 36: Giao diện thêm phòng

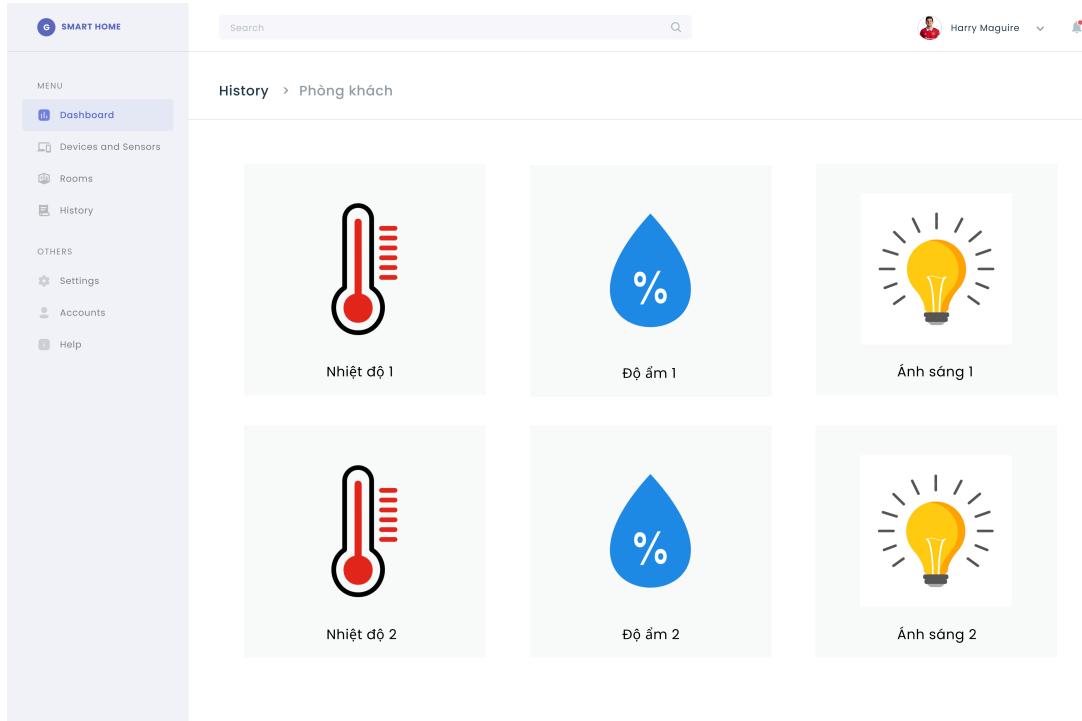


Hình 37: Giao diện chỉnh sửa thông tin phòng

6.2.5 Trang xem lịch sử



Hình 38: Giao diện chọn phòng



Hình 39: Giao diện chọn thiết bị



The screenshot shows a 'History' section for a room named 'Phòng khách'. The log table has two columns: 'Thời gian' (Time) and 'Nhiệt độ (độ C)' (Temperature). The data shows 10 entries all recorded at 25/02/2023 10:00:00, each with a value of 30.

Thời gian	Nhiệt độ (độ C)
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30
25/02/2023 10:00:00	30

Hình 40: Giao diện danh sách dữ liệu của thiết bị

6.3 Tổ chức thư mục trong dự án

Dự án được tổ chức thành các thư mục chính như sau:

- CodeHardware: Chứa code để khởi chạy việc kết nối phần cứng với server Adafruit - client: Thư mục hiện thực front-end của hệ thống, bao gồm các thư mục chính như sau:

- components: Chứa các component chung sử dụng cho hệ thống, có bao gồm các layout
- pages: Chứa các component của các trang trong hệ thống, trong folder này bao gồm các component sau:
 - Dashboard: Component về trang dashboard của hệ thống
 - Devices: Component về trang thiết bị của hệ thống
 - History: Component về trang lịch sử sử dụng thiết bị và cảm biến của hệ thống
 - Rooms: Component về trang danh sách các phòng của hệ thống
 - User: Chứa các component liên quan trực tiếp đến người dùng như: Đăng nhập, đăng ký, đổi mật khẩu và thông tin người dùng
- routes: Đây là thư mục để phân routes

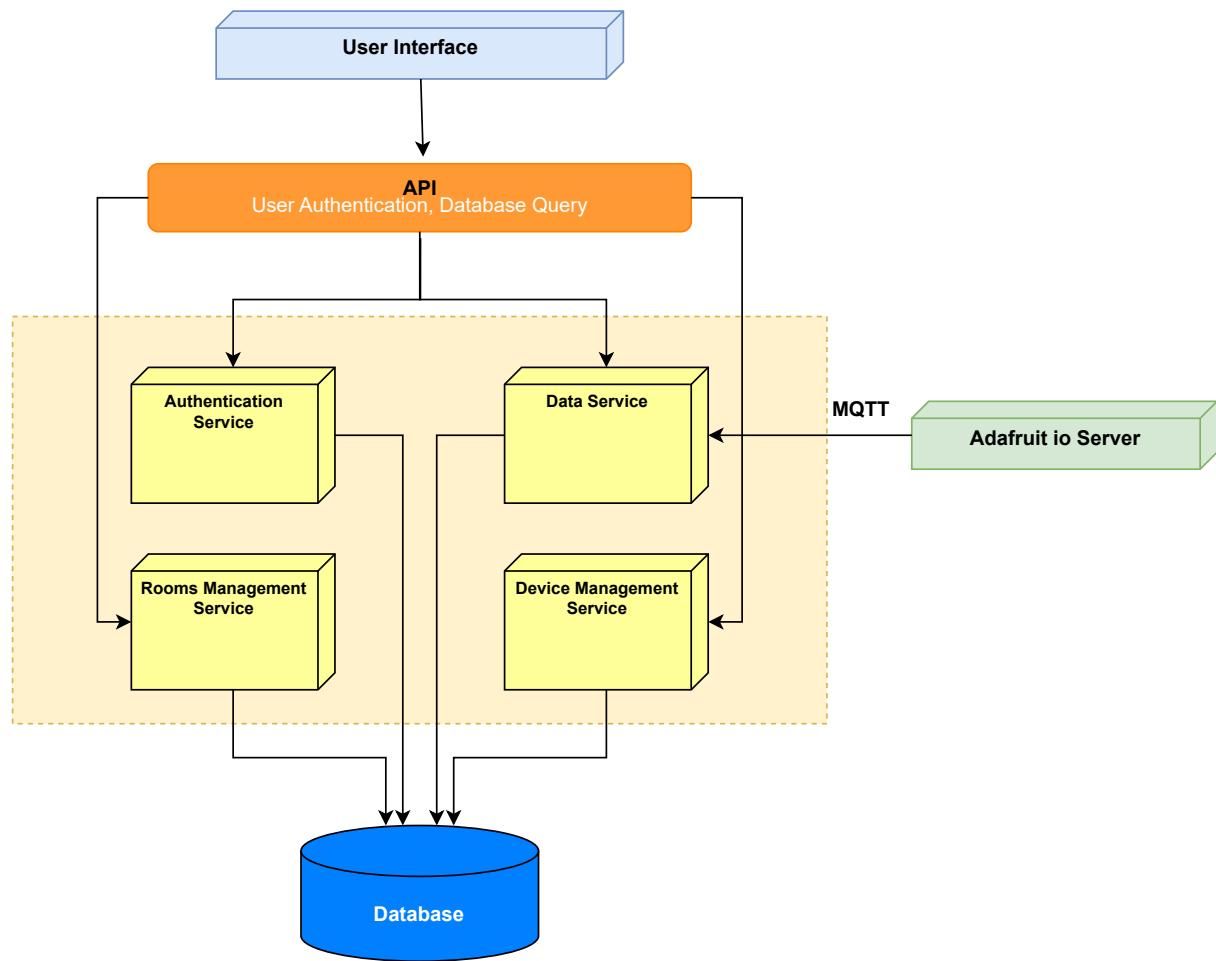
- server: Thư mục xử lý các sự kiện bên back-end, bao gồm các thư mục chính sau:

- controllers: Các controller để xử lý sự kiện tương tác giữa người dùng với hệ thống, gồm 2 file sau:
 - authControllers.js: Chứa các api xử lý sự kiện đăng nhập và đăng ký tài khoản người dùng



- recordControllers.js: Chứa các api xử lý các sự kiện khác nằm trong hệ thống
- db: Dùng để kết nối hệ thống với cơ sở dữ liệu
- models: Chứa các schema được sử dụng trong việc tương tác với cơ sở dữ liệu
- mqtt: Là nơi xử lý các dữ liệu nhận được từ server Adafruit theo thời gian thực
- routes: Các route trong Express được hiện thực và sử dụng bên phía client
- server.js: Nơi để khởi chạy tất cả sự kiện từ kết nối với server của cơ sở dữ liệu và Adafruit

6.4 Kiến trúc hệ thống



Hình 41: Lược đồ kiến trúc hệ thống

Trong đó người dùng(client) gửi các yêu cầu như là xem dữ liệu mới nhất của phòng, quản lý phòng, quản lý thiết bị, ... về cho server và server sẽ truy cập vào cơ sở dữ liệu để cung cấp dịch vụ cho người dùng. AdafruitIO cung cấp một server để lưu trữ và truyền dữ liệu giữa các thiết bị và các ứng dụng IoT của người dùng. Hệ thống sẽ sử dụng giao thức MQTT để lắng nghe và thực hiện việc lưu dữ liệu vào database khi server Adafruit cập nhật một dữ liệu mới. Lớp API nhằm cung cấp thêm một số dịch vụ tiện ích bổ sung như xác thực tài khoản, truy vấn database,..

Sau khi thiết kế được kiến trúc của phần mềm thì nhóm quyết định lựa chọn các công nghệ để hiện thực như sau:

- Frontend: ReactJS
- Backend: NodeJS
- Database: MongoDB

6.5 Công nghệ được sử dụng

6.5.1 ReactJS

ReactJS là một thư viện JavaScript mã nguồn mở được Facebook phát triển, đóng vai trò rất quan trọng trong việc xây dựng các ứng dụng web đơn trang (single-page applications) và các ứng dụng di động hiện nay. Với ReactJS, các nhà phát triển có thể xây dựng các thành phần (components) của ứng dụng và tái sử dụng chúng trong nhiều nơi khác nhau trong ứng dụng, giúp cải thiện sự tái sử dụng mã và hiệu suất. Điều này cũng giúp tăng tính linh hoạt và dễ bảo trì cho ứng dụng. ReactJS cũng cung cấp các công cụ để quản lý trạng thái (state) của ứng dụng, giúp cho việc tạo và quản lý các ứng dụng phức tạp trở nên dễ dàng hơn. Nó cho phép người phát triển thực hiện việc cập nhật giao diện người dùng một cách nhanh chóng và linh hoạt hơn, từ đó cải thiện trải nghiệm người dùng và hiệu quả hoạt động của ứng dụng. Bên cạnh đó, ReactJS còn có cộng đồng phát triển lớn và phổ biến, với rất nhiều tài liệu học tập và các công cụ hỗ trợ phát triển như React Developer Tools, Redux, React Native.



Hình 42: ReactJS

6.5.2 NodeJS

NodeJS là một nền tảng phát triển phần mềm được xây dựng trên engine JavaScript V8 của Google. Được phát triển bởi Ryan Dahl năm 2009, NodeJS đã nhanh chóng trở thành một trong những công nghệ phát triển web phổ biến nhất trên thế giới. NodeJS cho phép viết các ứng dụng back-end bằng JavaScript, đó là một ngôn ngữ phổ biến trong phát triển web front-end. Điều này giúp đơn giản hóa quá trình phát triển và tăng tính đồng nhất trong mã nguồn. NodeJS sử dụng mô hình non-blocking, sự kiện trên một luồng (event-driven, single-threaded) để xử lý các yêu cầu I/O. Điều này giúp giảm thời gian phản hồi và tăng hiệu suất xử lý của ứng dụng. NodeJS cũng có nhiều thư viện và framework hỗ trợ cho việc phát triển ứng dụng web, như Express, Koa, NestJS, ... Các thư viện này giúp đơn giản hóa và tăng tính linh hoạt trong việc phát triển ứng dụng. Ngoài ra, NodeJS còn có thể được sử dụng để xử lý các tác vụ đa luồng như xử lý dữ liệu, lập lịch, kết nối với cơ sở dữ liệu, ... NodeJS có thể hoạt động trên nhiều nền tảng, bao gồm Windows, macOS, Linux, ... Điều này giúp tăng tính di động của ứng dụng và tương thích trên nhiều hệ thống khác nhau. Nói chung, NodeJS là một công nghệ phát triển phần mềm mạnh mẽ và đa năng, đặc biệt là trong việc phát triển các ứng dụng web. Việc sử dụng NodeJS giúp cho quá trình phát triển ứng dụng trở nên dễ dàng hơn, đồng thời làm gia tăng hiệu suất và tính linh hoạt của ứng dụng.



Hình 43: NodeJS

6.5.3 ExpressJS

ExpressJS là một framework web được viết bằng JavaScript, cung cấp các công cụ và thư viện cho phát triển ứng dụng web back-end. Nó được xây dựng trên nền tảng NodeJS và có thể được sử dụng để phát triển các ứng dụng web độc lập hoặc kết hợp với các công nghệ khác. Với ExpressJS, việc xây dựng các API (Application Programming Interface) trở nên dễ dàng hơn. Nó cung cấp một cấu trúc tập trung trên route (đường dẫn) và middleware (phần mềm trung gian), cho phép người phát triển có thể tùy chỉnh các yêu cầu và phản hồi cho ứng dụng web. Các middleware có thể được sử dụng để xử lý các nhiệm vụ như xác thực, mã hóa, kiểm tra lỗi, xử lý dữ liệu và nhiều hơn nữa. ExpressJS cũng cho phép người phát triển quản lý các đối tượng và các tài nguyên như file, ảnh và video trên máy chủ của họ. Nó cũng có thể được sử dụng để kết nối với các cơ sở dữ liệu khác nhau, bao gồm MongoDB, MySQL và Postgres. ExpressJS có một cộng đồng phát triển rất lớn và sôi nổi, với nhiều tài liệu và ví dụ được chia sẻ trên các diễn đàn và trang web. Điều này giúp cho việc học và sử dụng ExpressJS trở nên dễ dàng hơn. Ngoài ra, ExpressJS còn được tích hợp với nhiều công nghệ khác như Socket.IO, PassportJS, GraphQL, ... Nó cũng hỗ trợ việc tạo ứng dụng web real-time thông qua tính năng WebSocket. Tóm lại, ExpressJS là một framework mạnh mẽ và linh hoạt cho phát triển ứng dụng web back-end. Nó cung cấp nhiều tính năng và công cụ để giúp người phát triển xây dựng các ứng dụng web nhanh chóng và dễ dàng hơn, đồng thời cũng giúp tăng tính bảo mật và hiệu suất của ứng dụng.



Hình 44: ExpressJS

6.5.4 MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) mã nguồn mở, được thiết kế để lưu trữ dữ liệu dưới dạng tài liệu (document) trong các bộ sưu tập (collection). Nó được phát triển bởi MongoDB Inc. và được phát hành lần đầu tiên vào năm 2009. MongoDB được thiết kế để xử lý các tài liệu có cấu trúc linh hoạt và phức tạp hơn so với các hệ quản trị cơ sở dữ liệu quan hệ truyền thống. Mỗi tài liệu trong MongoDB được lưu trữ dưới dạng JSON (JavaScript Object Notation), cho phép người dùng dễ dàng lưu trữ và truy xuất dữ liệu một cách linh hoạt. Với MongoDB, người dùng có thể lưu trữ và truy xuất dữ liệu một cách nhanh chóng và hiệu quả, đồng thời giảm thiểu thiết kế cơ sở dữ liệu phức tạp. Nó cung cấp các tính năng mạnh mẽ như replica set (bộ đôi sao chép) và sharding (phân tán), giúp tăng tính sẵn sàng, khả năng mở rộng và bảo mật của hệ thống. MongoDB cũng hỗ trợ nhiều ngôn ngữ lập trình như C++, Java, Python, Node.js, PHP, Ruby và nhiều ngôn ngữ khác, đồng thời cung cấp các API đa dạng như MongoDB Shell, MongoDB Compass, REST API và nhiều hơn nữa. Nó cung cấp tính năng tìm kiếm toàn văn (full-text search), aggregation framework (khung công việc tổng hợp) và map-reduce, giúp người dùng dễ dàng thực hiện các phép tính phức tạp trên dữ liệu. Tóm lại, MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ mạnh mẽ và linh hoạt, được thiết kế để lưu trữ và truy xuất dữ liệu dưới dạng tài liệu (document) trong các bộ sưu tập (collection). Nó cung cấp nhiều tính năng mạnh mẽ và đa dạng, giúp người dùng lưu trữ và truy xuất dữ liệu một cách hiệu quả và nhanh chóng.



Hình 45: MongoDB

6.5.5 Socket.IO

Socket.IO là một thư viện JavaScript mã nguồn mở được sử dụng để xây dựng các ứng dụng web thời gian thực với khả năng truyền tải dữ liệu hai chiều giữa máy chủ và trình duyệt. Thư viện này cung cấp một phương tiện đơn giản, hiệu quả và linh hoạt để truyền tải thông tin từ máy chủ đến trình duyệt và ngược lại. Socket.IO sử dụng giao thức WebSocket, một giao thức truyền tải dữ liệu hai chiều thông qua một kết nối duy nhất giữa máy chủ và trình duyệt. Nó hỗ trợ nhiều loại trình duyệt và thiết bị, bao gồm cả máy tính để bàn, điện thoại di động và máy tính bảng. Với Socket.IO, chúng ta có thể xây dựng các ứng dụng có tính tương tác cao như trò chơi trực tuyến, ứng dụng trò chuyện và cộng tác thời gian thực. Thông qua việc sử dụng socket, ứng dụng của chúng ta có thể truyền tải dữ liệu và nhận được phản hồi chỉ trong vài mili giây, cung cấp cho người dùng trải nghiệm ứng dụng nhanh và mượt hơn. Với cách thức hoạt động đơn giản, Socket.IO cho phép các nhà phát triển tập trung vào việc xây dựng ứng dụng thời gian thực một cách dễ dàng hơn. Nó cung cấp cho nhà phát triển khả năng để tạo kết nối giữa máy chủ

và trình duyệt và quản lý các sự kiện trong ứng dụng của họ bằng cách sử dụng JavaScript. Tóm lại, Socket.IO là một công cụ quan trọng giúp tạo ra các ứng dụng web thời gian thực hiệu quả và linh hoạt. Thư viện này cung cấp một công cụ đơn giản để truyền tải dữ liệu hai chiều và đáp ứng nhanh chóng, giúp nâng cao trải nghiệm người dùng.



Hình 46: Socket.IO

6.5.6 JSON Web Token

JSON Web Token (JWT) là một chuẩn mã hóa dữ liệu trong các ứng dụng web. JWT cho phép truyền tải thông tin giữa các bên dưới dạng một chuỗi JSON được mã hóa. Điều này đảm bảo tính toàn vẹn của dữ liệu và ngăn chặn sự thay đổi trái phép khi thông tin được truyền tải. JWT được sử dụng rộng rãi trong việc xác thực người dùng, nó cho phép máy chủ xác minh danh tính của người dùng một cách an toàn và hiệu quả. JWT bao gồm ba phần chính: header, payload và signature. Header chứa thông tin về loại mã hóa được sử dụng. Payload chứa thông tin người dùng và các thuộc tính liên quan đến người dùng. Signature được tạo ra từ header, payload và một secret key, và được sử dụng để xác minh tính toàn vẹn của JWT. Sử dụng JWT có nhiều lợi ích, bao gồm tính linh hoạt và thông dụng. JWT có thể được sử dụng trên bất kỳ nền tảng nào và có thể được tích hợp vào các ứng dụng web khác nhau. Nó cũng cho phép lưu trữ thông tin người dùng một cách an toàn và bảo mật, giúp tăng tính bảo mật của ứng dụng web. Tuy nhiên, JWT cũng có những hạn chế, bao gồm kích thước tệp lớn hơn so với các phương thức xác thực khác và không thể thu hồi được token đã phát hành. Tổng quan, JWT là một công nghệ mã hóa dữ liệu phổ biến trong việc xác thực người dùng và truyền tải thông tin giữa các bên trong các ứng dụng web. JWT cho phép lưu trữ thông tin người dùng một cách an toàn và hiệu quả, đồng thời tăng tính bảo mật của ứng dụng web.



Hình 47: JSON Web Token

6.5.7 Formik

Formik là một thư viện quản lý biểu mẫu được sử dụng phổ biến trong React. Với Formik, việc xây dựng các biểu mẫu trở nên đơn giản hơn nhờ vào tính năng tự động quản lý trạng thái của biểu mẫu và các thành phần liên quan. Formik cung cấp cho người dùng nhiều tính năng hỗ trợ, bao gồm: kiểm tra đầu vào và xác thực dữ liệu, hỗ trợ nhiều loại input khác nhau, tích hợp với các thư viện UI như Material UI hay Bootstrap, và tích hợp với Yup để xác thực dữ liệu trên client-side. Với sự hiệu quả và tính linh hoạt của Formik, nhà phát triển có thể tập trung vào việc xây dựng các ứng dụng phức tạp và tương tác cao mà không phải lo lắng về việc quản lý biểu mẫu. Kết hợp với sức mạnh của React, Formik là một công cụ quan trọng để tạo ra các ứng dụng web chất lượng cao và đáp ứng nhu cầu người dùng.



Hình 48: Formik

6.6 Sản phẩm hiện thực

6.6.1 Design Pattern

Design Pattern mà nhóm sử dụng là Singleton Pattern. Việc sử dụng design pattern này để đảm bảo class Database chỉ có duy nhất một instance và cung cấp một cách truy cập đến instance đó.

```
1  const { MongoClient } = require("mongodb");
2  const Db = process.env.ATLAS_URI;
3  const client = new MongoClient(Db, {
4      useNewUrlParser: true,
5      useUnifiedTopology: true,
6  });
7
8  let instance = null;
9
10 You, 7 hours ago | 1 author (You)
11 class Database {
12     constructor() {
13         if (!instance) {
14             client.connect((err, db) => {
15                 if (db) {
16                     this.db = db.db("smarthome");
17                     console.log("Successfully connected to MongoDB.");
18                 }
19             });
20             instance = this;
21         }
22         return instance;
23     }
24     getDb() {
25         return this.db;
26     }
27 }
28
29 module.exports = new Database();
```

Hình 49: Hiện thực kết nối với MongoDB sử dụng Singleton Pattern

```
server.listen(port, () => {
    // perform a database connection when server starts
    dbo.getDb();

    console.log(`Server is running on port: ${port}`);

    mqtt.subscribe((err) => {
        if (err) console.error(err);
    });
});
```

Hình 50: Gọi getDB() để tạo instance Database và sử dụng cho sau này

6.6.2 API documentation

Để có thể tạo ra một bản tài liệu cho các API đã thiết kế, nhóm sử dụng Swagger, một thư viện giúp tạo ra một UI cho việc kiểm tra và hiển thị chi tiết của API. Để có thể truy cập, đầu tiên cần phải khởi chạy server bằng cách gõ:

- cd src
- nmp install
- npm start

Sau đó, mở trình duyệt web và truy cập đường link: <http://localhost:5000/api-docs/#/>. Giao diện sẽ được hiển thị như sau:



Swagger
powered by SMARTBEAR

Smart home API 1.0.0 OAS3

An example API for demonstration purposes

Servers
http://localhost:5000 - Smart home server ▾

Temp API endpoints for get latest temperature data

GET /api/temp Get the latest temperature data

Humi API endpoints for get latest humidity data

GET /api/humi Get the latest humidity data

Light API endpoints for get latest light data

GET /api/light Get the latest light data

POST /api/light Add light's data to adafruit server

Fan API endpoints for get latest fan data

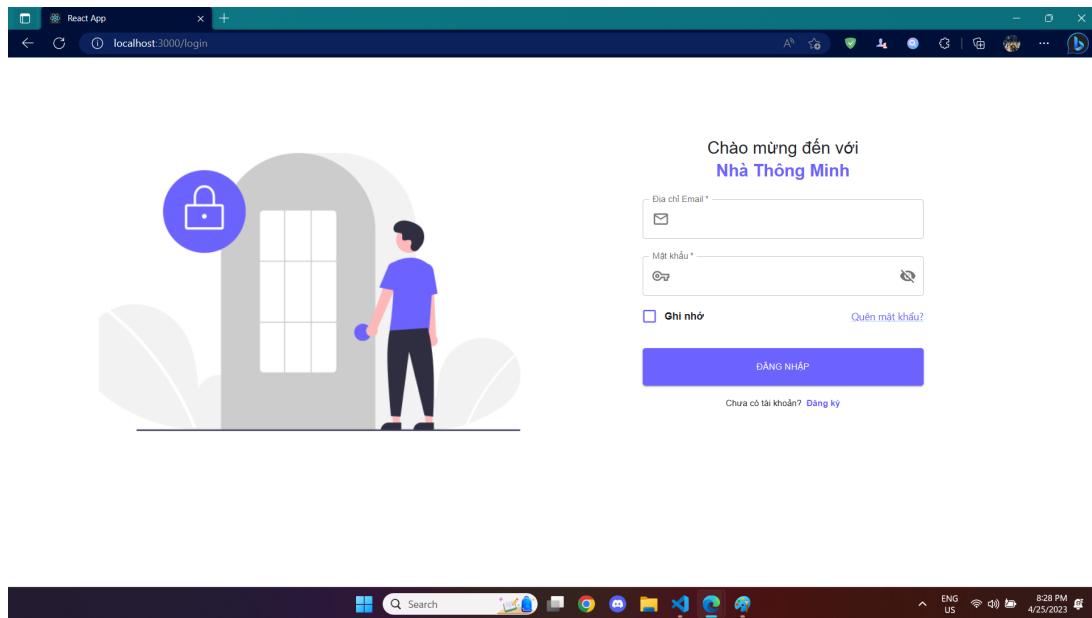
GET /api/fan Get the latest fan data

POST /api/fan Add fan's data to adafruit server

Hình 51: Giao diện Swagger UI

6.6.3 Kết quả hiện thực trang web

- Trang đăng nhập, đăng ký tài khoản: Đây là trang để người dùng tạo tài khoản và đăng nhập vào hệ thống



Hình 52: Giao diện trang đăng nhập hệ thống



Chào mừng đến với
Nhà Thông Minh

Địa chỉ email *

Mật khẩu *

Xác nhận mật khẩu *

Tôi đã đọc và đồng ý với điều khoản và điều kiện

ĐĂNG KÝ

Đã có tài khoản? [Đăng nhập](#)



Hình 53: Giao diện trang đăng ký tài khoản

Chào mừng đến với
Nhà Thông Minh

Địa chỉ email *

Mật khẩu *

Xác nhận mật khẩu *

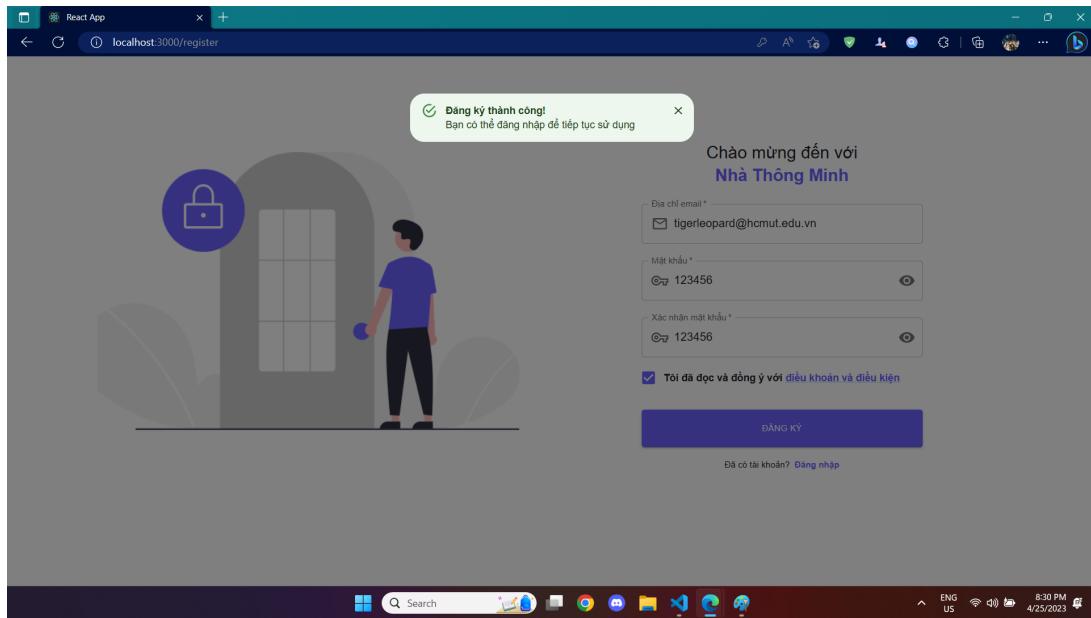
Tôi đã đọc và đồng ý với điều khoản và điều kiện

ĐĂNG KÝ

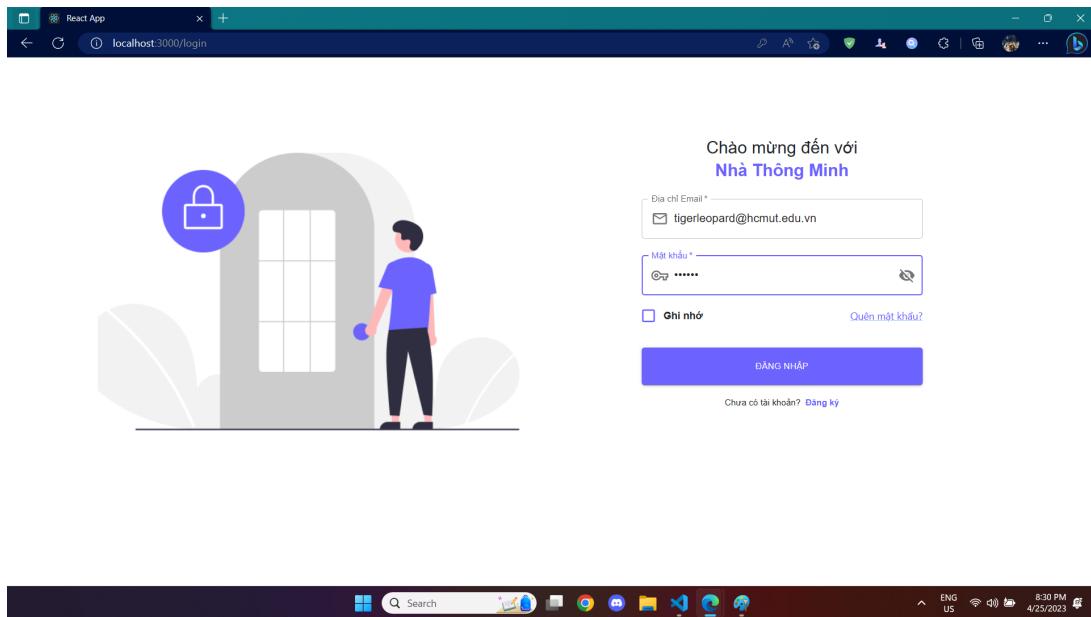
Đã có tài khoản? [Đăng nhập](#)



Hình 54: Người dùng điền thông tin để tạo tài khoản

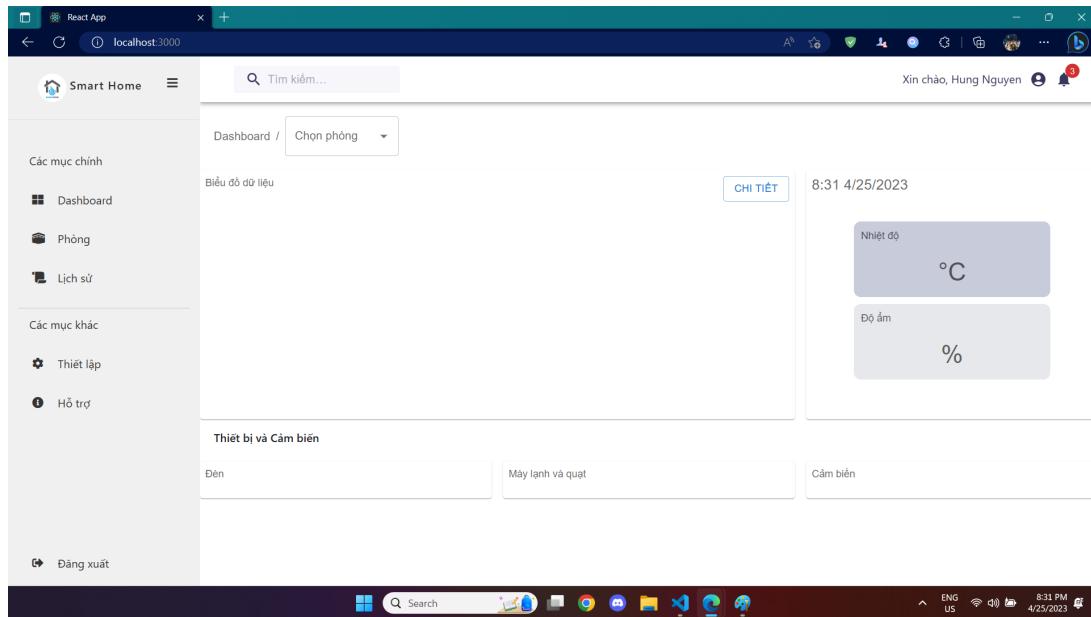


Hình 55: Chọn Đăng ký và hệ thống sẽ xác nhận tài khoản mới đã được tạo

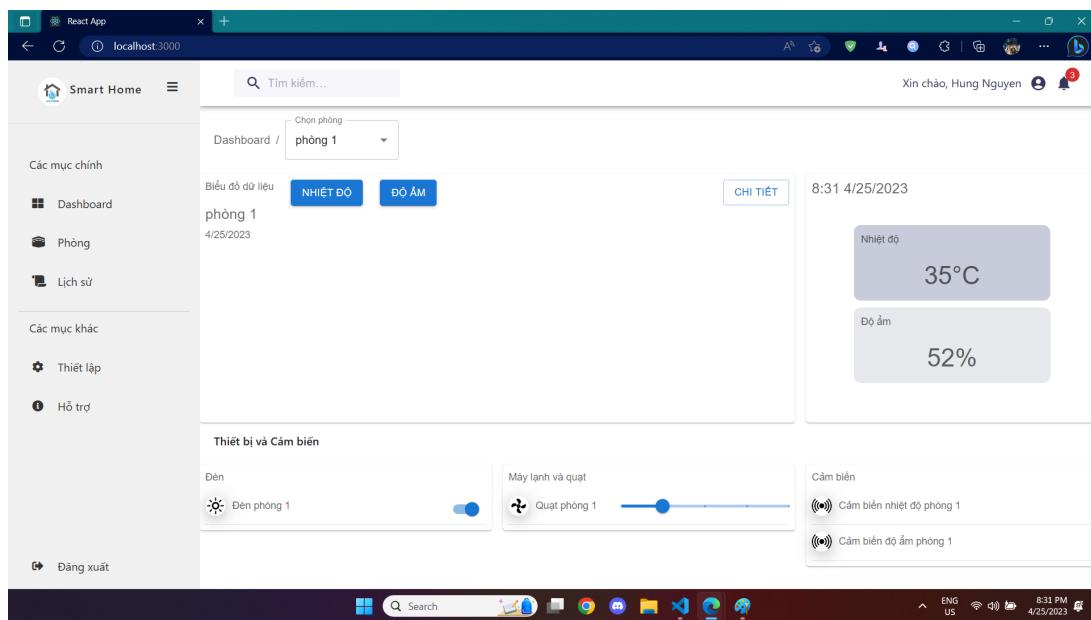


Hình 56: Quay trở lại trang đăng nhập để đăng nhập tài khoản vừa tạo

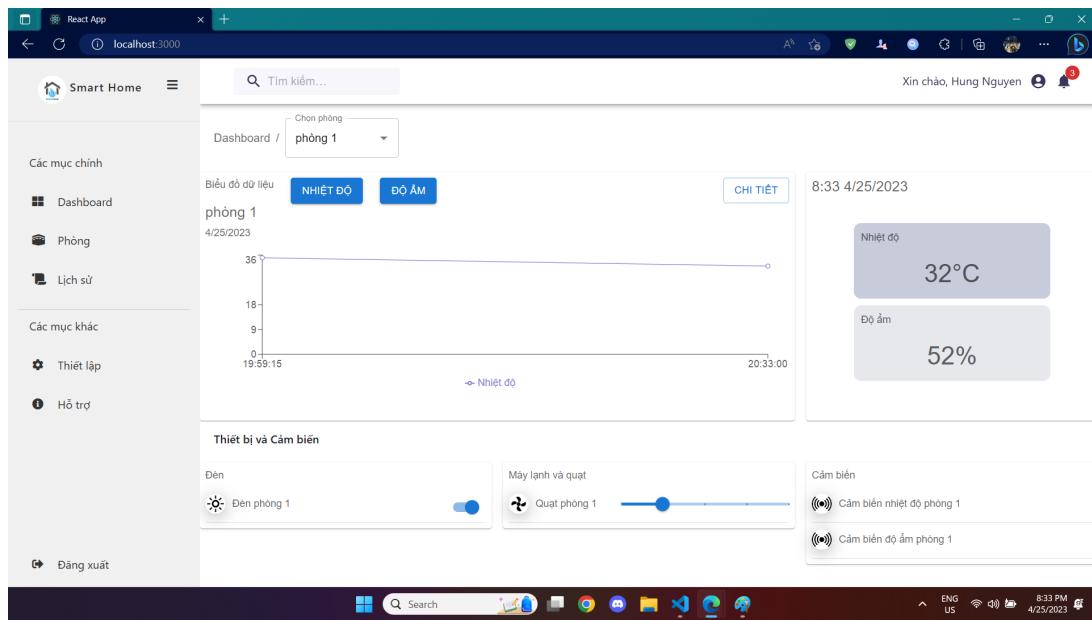
- Trang dashboard: Đây là trang để hiển thị dữ liệu được trực quan hóa từ các cảm biến và cho phép người dùng điều khiển các thiết bị quạt, đèn



Hình 57: Giao diện trang dashboard

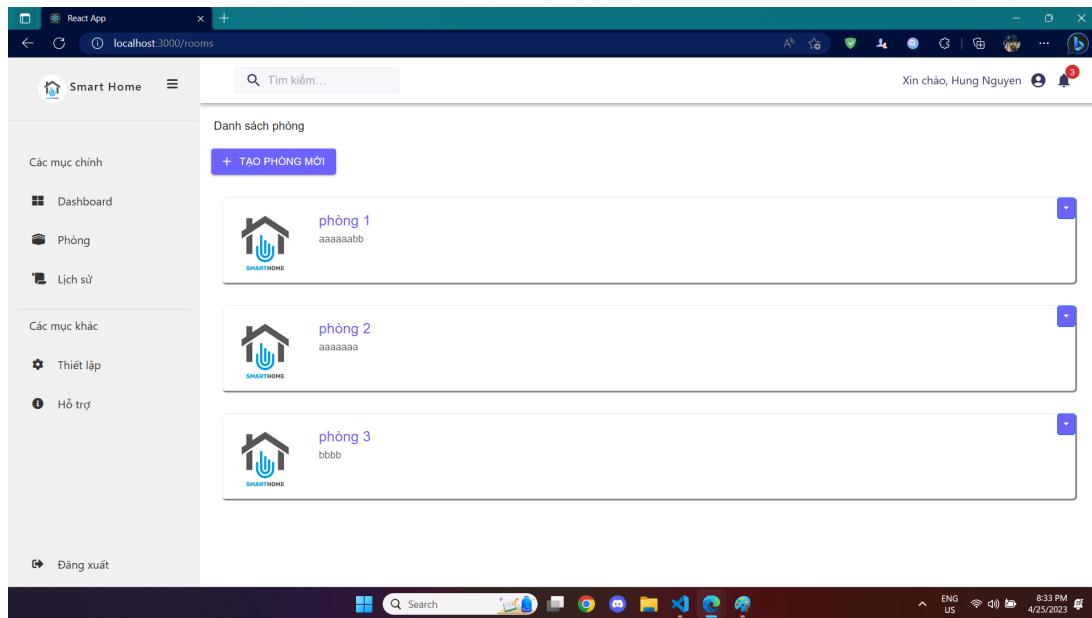


Hình 58: Người dùng chọn phòng để hiển thị các thiết bị có thể chọn

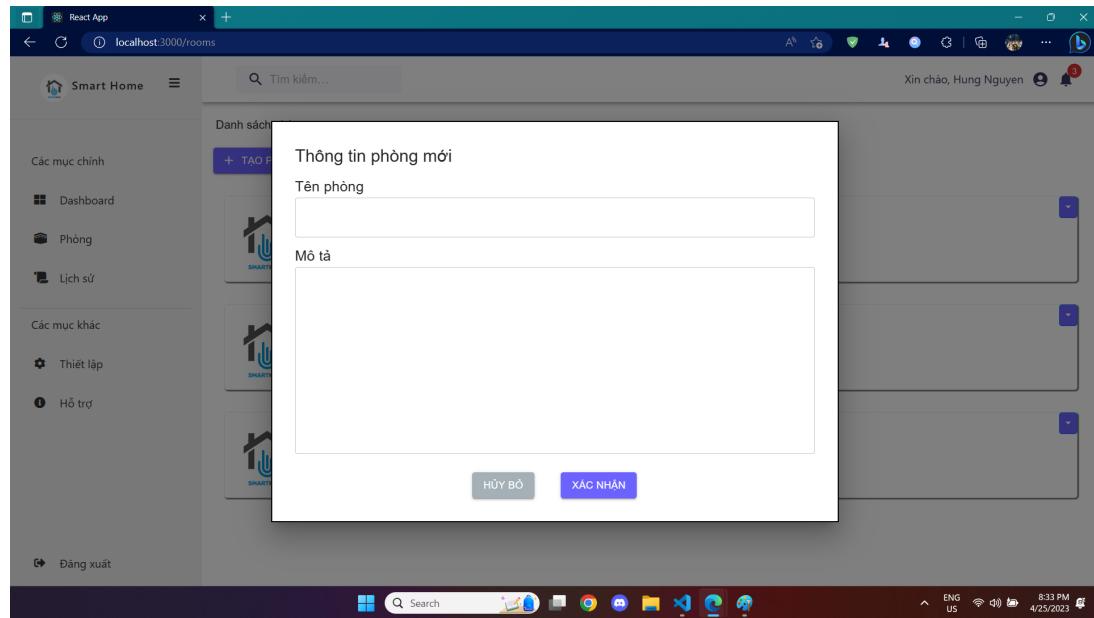


Hình 59: Sau khi chọn thiết bị thì một đồ thị dữ liệu sẽ xuất hiện

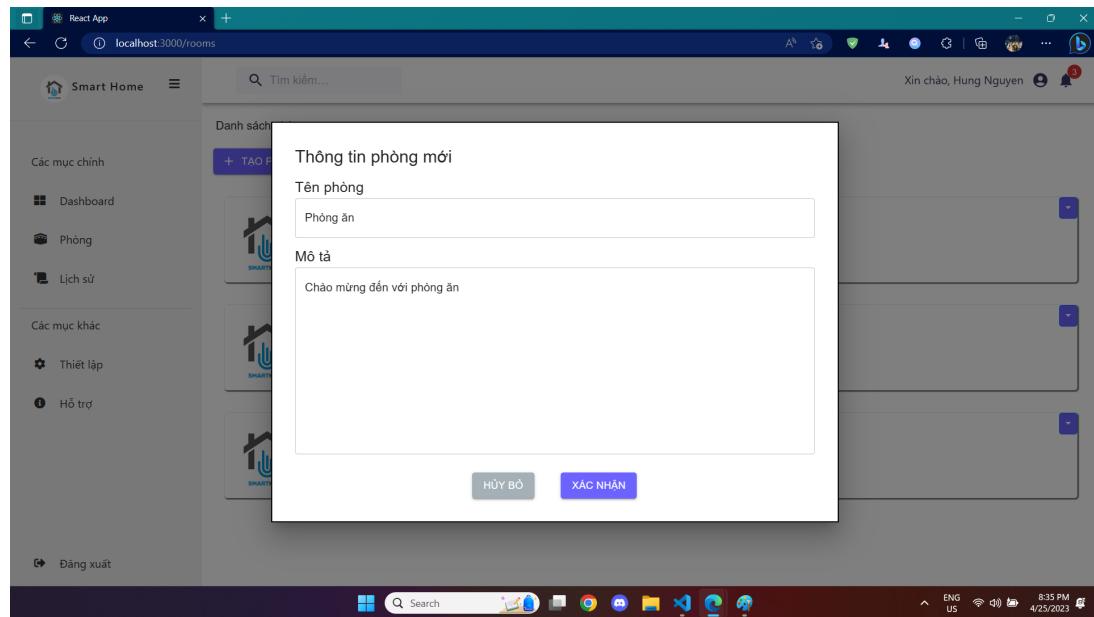
- Trang danh sách phòng: Người dùng có thể tạo phòng mới, chỉnh sửa thông tin phòng và xóa phòng. Đồng thời khi nhấp vào từng phòng thì sẽ dẫn đến trang danh sách thiết bị và cảm biến của phòng đó.



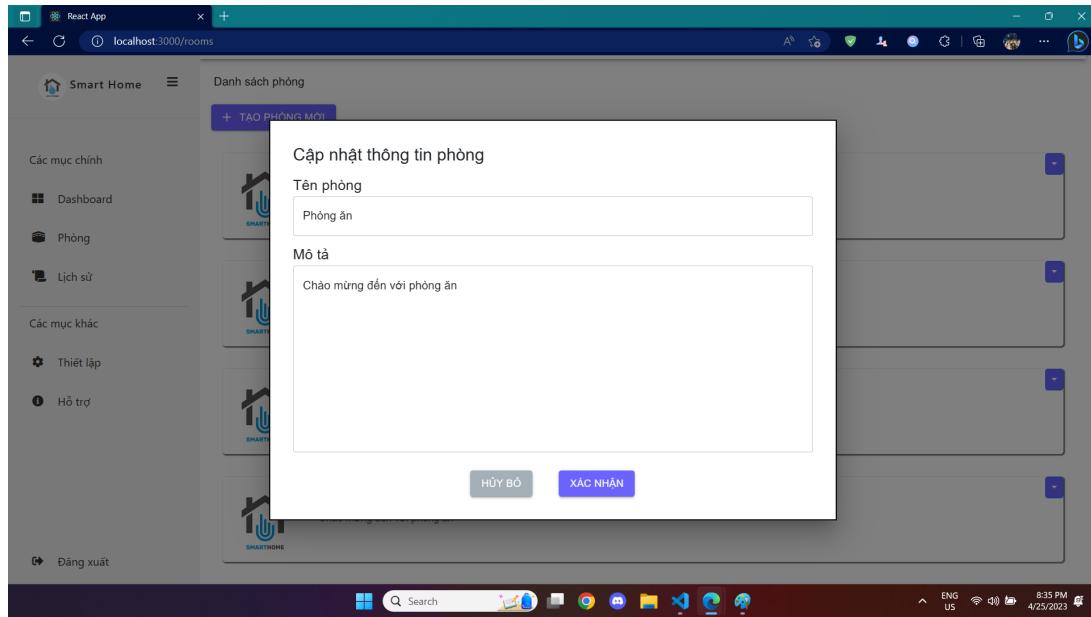
Hình 60: Giao diện trang danh sách phòng



Hình 61: Người dùng chọn Tạo phòng mới, sau đó điền thông tin

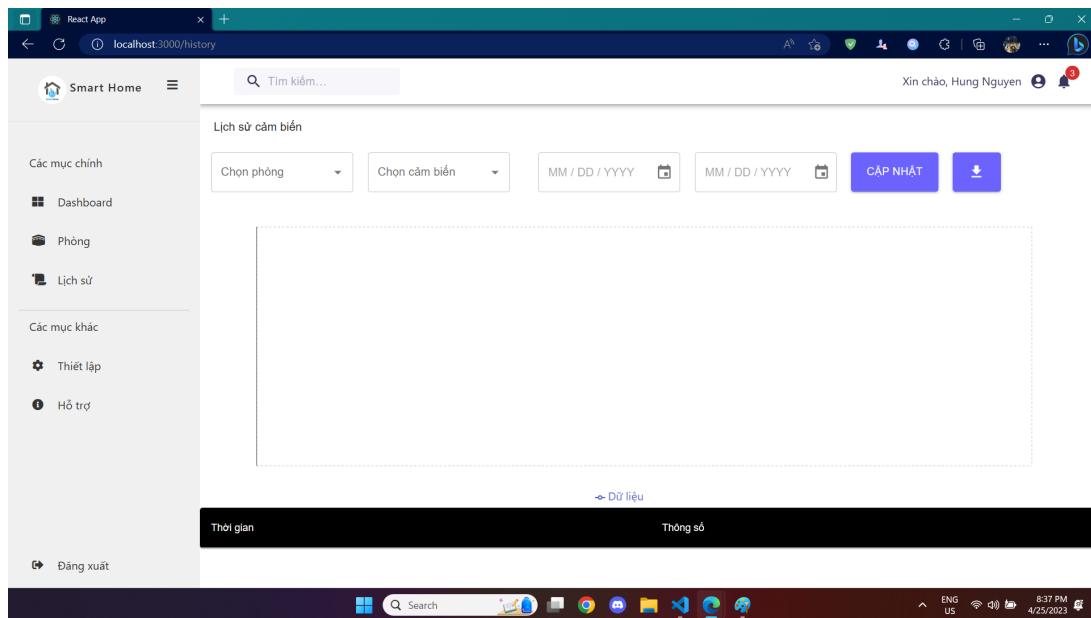


Hình 62: Khi xác nhận thêm phòng mới, nó sẽ được hiển thị lên danh sách

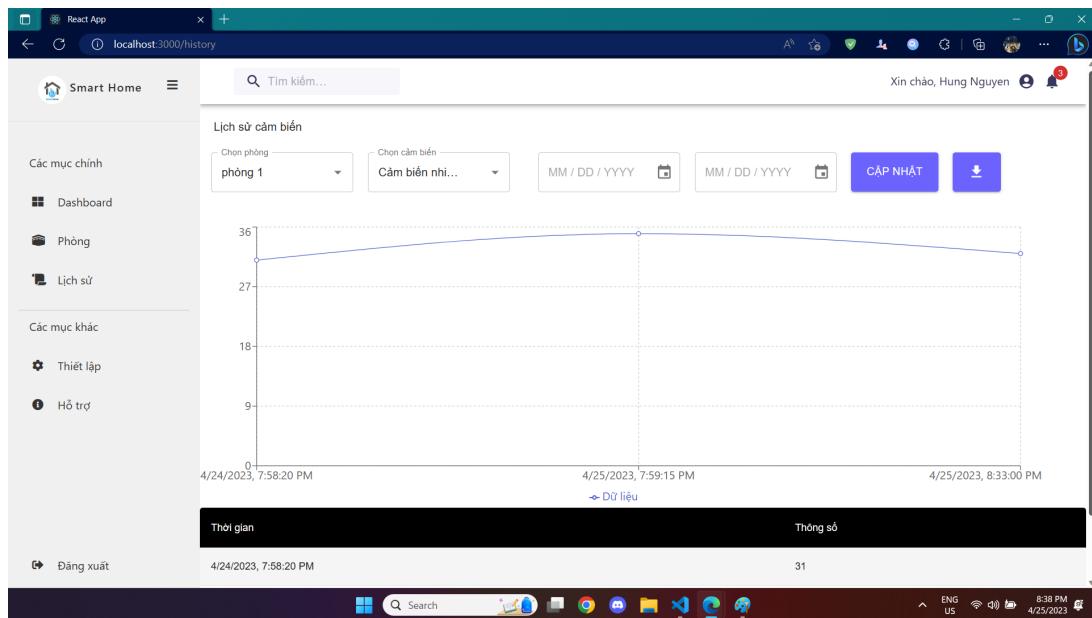


Hình 63: Người dùng cũng có thể chỉnh sửa thông tin của phòng

Trang lịch sử: Người dùng có thể xem được lịch sử dữ liệu của thiết bị và cảm biến dưới dạng bảng và đồ thị, đồng thời có thể tải dữ liệu về dưới dạng file csv.



Hình 64: Giao diện trang lịch sử



Hình 65: Sau khi chọn phòng, thiết bị và chọn Cập nhật thì bảng và đồ thị dữ liệu sẽ xuất hiện

The screenshot shows a Microsoft Excel spreadsheet titled "tigerleopard (2).csv". The table has two columns: "Time" and "Message". The data is as follows:

	Time	Message
1	4/24/2023 19:58	31
2	4/25/2023 19:59	35
3	4/25/2023 20:33	32
4		
5		
6		
7		

Hình 66: Khi người dùng chọn tải dữ liệu về thì sẽ nhận được file csv như sau

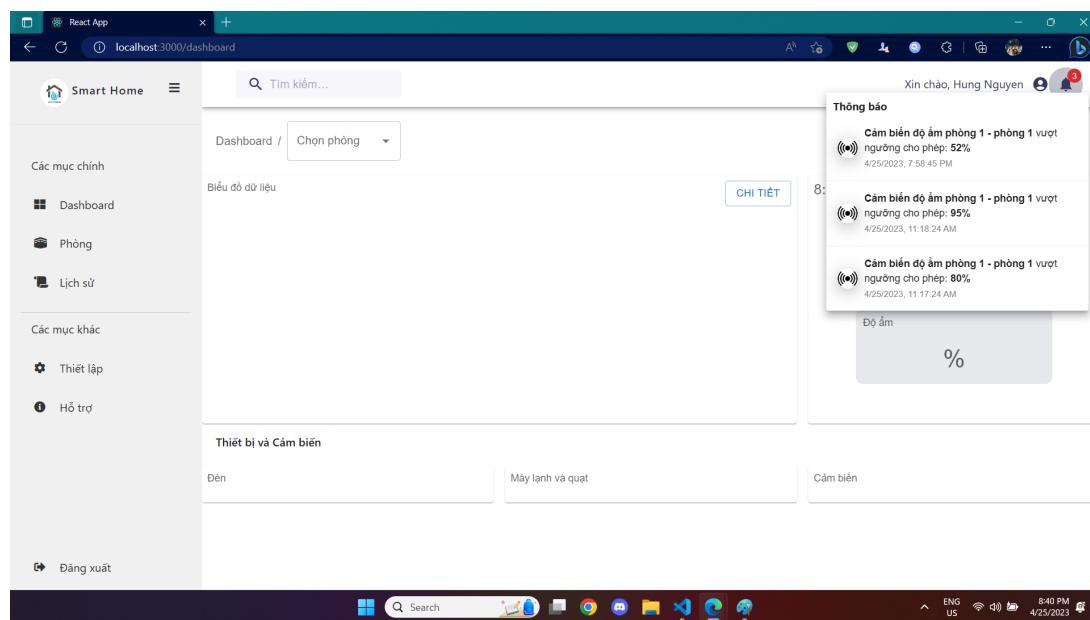


- Trang thông tin người dùng: Người dùng có thể cập nhật thông tin cá nhân của mình tại đây

The screenshot shows a user profile page titled "Thông tin người dùng". It includes a profile picture of a man, a button to "ĐỔI ANH ĐẠI DIỆN" (Change Photo), and several input fields for personal information: Họ và tên*, Giới tính*, Ngày sinh*, Số điện thoại*, Địa chỉ*. At the bottom is a blue "XÁC NHẬN" (Confirm) button.

Hình 67: Giao diện trang thông tin cá nhân

- Thanh thông báo: Người dùng sẽ nhận được thông báo khi thông số của cảm biến mà hệ thống nhận được vượt ngưỡng quy định do người dùng thiết lập.



Hình 68: Thanh thông báo khi có sự vượt ngưỡng dữ liệu

6.6.4 Mã nguồn của nhóm

Để truy cập vào mã nguồn của nhóm, vui lòng vào đường dẫn sau: <https://github.com/DanhNguyen02/SmartHome> hoặc quét QR Code dưới đây



Hình 69: Quét QR Code để nhận link github mã nguồn của nhóm



Tài liệu

[1] Introduction to NodeJS

<https://nodejs.dev/en/learn/>

[2] What is MERN Stack?

<https://www.educative.io/answers/what-is-mern-stack>

[3] Node.JS and Singleton Pattern

<https://medium.com/swlh/node-js-and-singleton-pattern-7b08d11c726a>

[4] Socket.IO: What it is, when to use it, and how to get started

<https://ably.com/topic/socketio>

[5] What is JWT? How Does It Work?

<https://www.akana.com/blog/what-is-jwt>

[6] Swagger and Swagger UI: What is it and Why is it a must for your APIs?

<https://www.chakray.com/swagger-and-swagger-ui-what-is-it-and-why-is-it-a-must-for-yourapis/>