

5. Kiến trúc ứng dụng di động

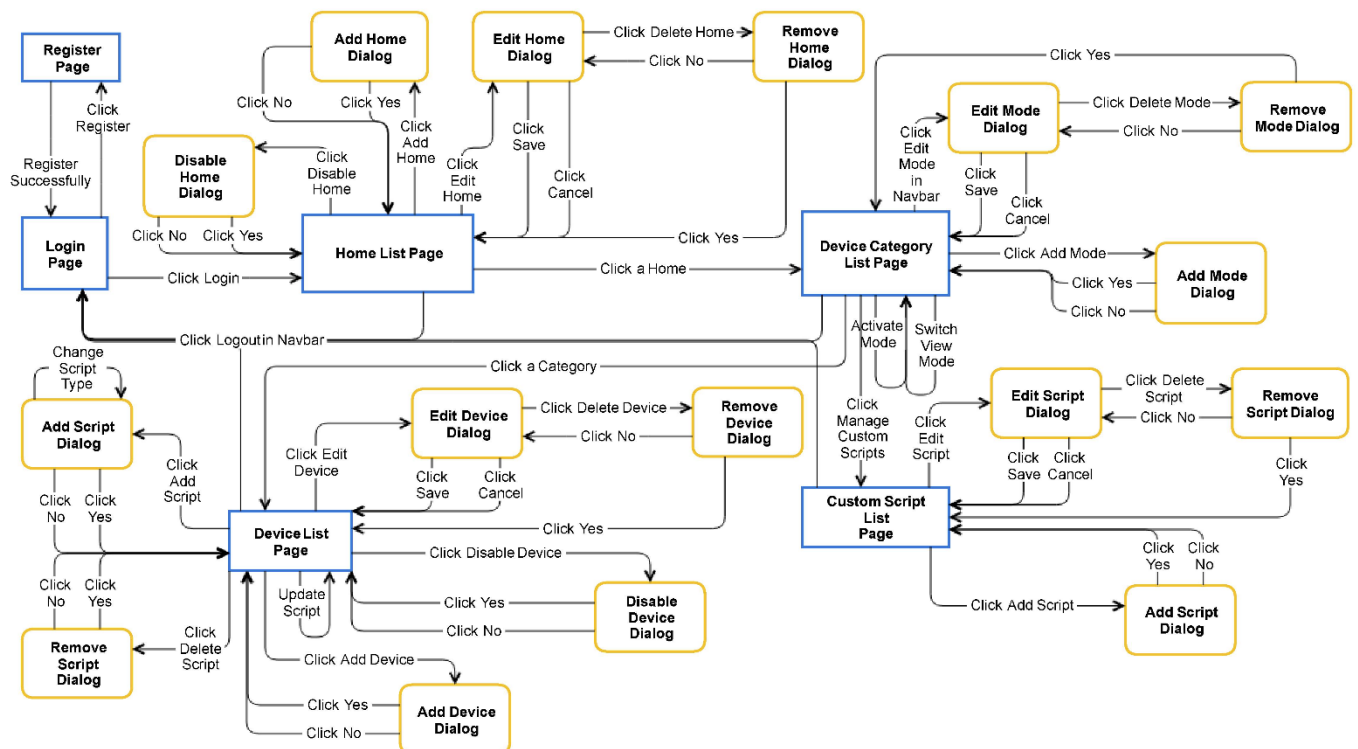
5.1. Các chức năng của ứng dụng

Ứng dụng di động nhà thông minh được thiết kế để hỗ trợ người dùng dễ dàng:

- Đăng ký và đăng nhập vào hệ thống
- Quản lý các ngôi nhà được lắp đặt hệ thống
- Quản lý các Chế độ người dùng tự tạo của ngôi nhà
- Quản lý các thiết bị hệ thống hỗ trợ được lắp đặt vào nhà
- Quản lý các Kịch bản định sẵn của từng thiết bị
- Quản lý các Kịch bản người dùng tự tạo của ngôi nhà

5.2. Kiến trúc tổng quát

Sơ đồ luồng Giao diện của ứng dụng di động:



Hình 5.1: Sơ đồ luồng Giao diện của ứng dụng di động

Ứng dụng có kiến trúc bao gồm 6 Giao diện (Page) chính là Giao diện Đăng nhập, Giao diện Đăng ký, Giao diện Danh sách các ngôi nhà, Giao diện Danh sách các kiểu thiết bị, Giao diện Danh sách các thiết bị và Giao diện Danh sách các kịch bản tự tạo. Trong mỗi Giao diện chính có các Bảng hộp thoại (Dialog) hỗ trợ người dùng thực hiện các chức năng của ứng dụng.

Giao diện Đăng ký (Register Page): Giao diện cho phép người dùng đăng ký tài khoản hệ thống với các thông tin là Tên đầy đủ, Tên đăng nhập, Mật khẩu, Xác nhận mật khẩu và địa chỉ Email. Sau khi người dùng nhập đầy đủ các thông tin và gửi lên hệ thống, một Email sẽ được gửi tới địa chỉ Email của người dùng để xác nhận các thông tin đăng ký. Giao diện này không có Bảng hộp thoại nào. Từ Giao diện này, người dùng có thể đến Giao diện đăng nhập sau khi đã gửi thông tin đăng ký.

Giao diện Đăng nhập (Login Page): Giao diện cho phép người dùng đăng nhập vào hệ thống với tên tài khoản và mật khẩu mà người dùng đã đăng ký. Giao diện này không có Bảng hộp thoại nào. Từ giao diện này người dùng có thể đi đến Giao diện đăng ký bằng cách nhấn nút Đăng ký (Register) hoặc đến Giao diện Danh sách các ngôi nhà sau khi đã đăng nhập thành công.

Giao diện Danh sách các ngôi nhà (Home List Page): Danh sách tên các ngôi nhà được lắp đặt hệ thống của người dùng được thể hiện ở Giao diện này. Với mỗi ngôi nhà, khi người dùng nhấn nút chỉnh sửa, Bảng hộp thoại Chỉnh sửa ngôi nhà sẽ hiện ra, tại đây người dùng có thể xem đầy đủ cũng như cập nhật lại thông tin của ngôi nhà, bao gồm Tên, Địa chỉ và Mô tả. Để xóa một ngôi nhà, người dùng có thể nhấn nút Xóa (Remove Home) tại Bảng hộp thoại này, một Bảng hộp thoại khác hiện lên yêu cầu người dùng xác nhận thao tác Xóa. Để thêm mới một ngôi nhà, người dùng có thể nhấn nút Thêm mới (Add Home) tại Giao diện chính để mở Bảng hộp thoại Thêm mới yêu cầu người dùng nhập đầy đủ các thông tin về nhà mới. Ngoài ra, người dùng còn có thể dừng hoạt động của hệ thống hoặc cho phép hệ thống hoạt động trở lại tại mỗi ngôi nhà thông qua việc nhấn nút Tắt/Mở (Disable/Enable). Từ giao diện này, khi nhấn Đăng xuất (Logout) từ thanh công cụ, người dùng sẽ về Giao diện Đăng nhập hoặc khi nhấn vào một ngôi nhà, người dùng sẽ tới Giao diện Danh sách các kiểu thiết bị.

Giao diện Danh sách các kiểu thiết bị (Device Category List Page): Danh sách tên các kiểu thiết bị và Danh sách tên các chế độ thuộc ngôi nhà được chọn sẽ được thể hiện ở Giao diện này. Tại đây, người dùng có thể chuyển đổi Chế độ (Mode) thông qua hộp trình đơn thả xuống (Dropdown box) hoặc kích hoạt (Activate) một Chế độ bất kì bằng cách nhấn vào tên Chế độ đó tại thanh công cụ. Để chỉnh sửa thông tin hoặc xóa một Chế độ nào đó, người dùng nhấn vào nút chỉnh sửa cạnh tên Chế độ đó tại thanh công cụ. Để thêm mới một Chế độ, người dùng có thể nhấn nút Thêm mới (Add Mode) tại giao diện chính để mở Bảng hộp thoại Thêm mới yêu cầu người dùng nhập đầy đủ các thông tin về Chế độ mới. Từ giao diện này, khi nhấn Đăng xuất (Logout) từ thanh công cụ, người dùng sẽ về Giao diện Đăng nhập, khi nhấn vào một kiểu thiết bị, người dùng sẽ tới Giao diện Danh sách các thiết bị hoặc khi nhấn vào nút Quản lý Kịch bản tự tạo (Manage Custom Scripts), người dùng sẽ tới Giao diện Danh sách các Kịch bản tự tạo.

Giao diện Danh sách các thiết bị (Device List Page): Danh sách tên, chân GPIO các thiết bị thuộc kiểu thiết bị được chọn cũng như Danh sách các Kịch bản định sẵn thuộc Chế độ được chọn của từng thiết bị được thể hiện ở Giao diện này. Với mỗi thiết bị, khi người dùng nhấn nút chỉnh sửa, Bảng hộp thoại Chỉnh sửa thiết bị sẽ hiện ra, tại đây người dùng có thể xem đầy đủ cũng như cập nhật lại thông tin của thiết bị, bao gồm Tên, Mô tả và Vị trí. Để xóa một thiết bị, người dùng có thể nhấn nút Xóa (Remove Device) tại Bảng hộp thoại này, một Bảng hộp thoại khác hiện lên yêu cầu người dùng xác nhận thao tác Xóa. Để thêm mới một thiết bị, người dùng có thể nhấn nút Thêm mới (Add Device) tại Giao diện chính để mở Bảng hộp thoại Thêm mới yêu cầu người dùng nhập đầy đủ các thông tin về thiết bị mới. Danh sách các Kịch bản thuộc một thiết bị sẽ được thể hiện khi người dùng nhấn vào thiết bị đó, người dùng có thể Xóa Kịch bản bằng cách nhấn nút Xóa ở đầu mỗi Kịch bản hoặc cập nhật lại thông tin Kịch bản như Điều kiện, Hành động hay Thời gian bắt đầu, Thời gian kết thúc thông qua các hộp trình đơn thả xuống. Ngoài ra để thêm mới một Kịch bản, người dùng có thể nhấn nút Thêm mới Kịch bản (Add Script), Bảng hộp thoại Thêm mới Kịch bản hiện ra, người dùng cần chọn Loại Kịch bản muốn thêm và xác định nội dung cho kịch bản đó. Từ giao diện này, khi nhấn Đăng xuất (Logout) từ thanh công cụ, người dùng sẽ về Giao diện Đăng nhập hoặc khi nhấn vào biểu tượng Ngôi nhà (Home) trên thanh công cụ, người dùng sẽ quay lại Giao diện Danh sách các kiểu thiết bị.

Giao diện Danh sách các Kịch bản tự tạo (Custom Script List Page): Danh sách tên các Kịch bản người dùng tự tạo thuộc Chế độ được chọn sẽ được thể hiện ở Giao diện này. Với mỗi Kịch bản tự tạo, khi người dùng nhấn nút chỉnh sửa, Bảng hộp thoại Chỉnh sửa Kịch bản tự tạo sẽ hiện ra, tại đây người dùng có thể xem cũng như cập nhật lại nội dung và tên của kịch bản. Để thêm mới một Kịch bản tự tạo, người dùng có thể nhấn nút Thêm mới Kịch bản tự tạo (Add Custom Script), Bảng hộp thoại Thêm mới Kịch bản tự tạo hiện ra yêu cầu người dùng xác định tên và nội dung cho kịch bản mới. Từ giao diện này, khi nhấn Đăng xuất (Logout) từ thanh công cụ, người dùng sẽ về Giao diện Đăng nhập hoặc khi nhấn vào biểu tượng Ngôi nhà (Home) trên thanh công cụ, người dùng sẽ quay lại Giao diện Danh sách các kiểu thiết bị.

Với cách thiết kế như trên, kiến trúc ứng dụng di động cung cấp một giao diện tương đối đơn giản nhưng vẫn đáp ứng được các nhu cầu cơ bản về việc quản lý các ngôi nhà, thiết bị, chế độ và các kịch bản điều khiển thiết bị của người dùng.

6. Hiện thực và đánh giá

6.3. Phát triển ứng dụng di động

6.3.1 Hướng phát triển ứng dụng di động

Hiện nay có 2 xu hướng phát triển ứng dụng di động chính là: Phát triển ứng dụng di động thuần túy (Native Mobile Application Development) và Phát triển ứng dụng di động lai (Hybrid Mobile Application Development).

Phát triển ứng dụng di động thuần túy (Native Mobile Application Development)

Phát triển ứng dụng di động thuần túy là phát triển một ứng dụng đặc biệt chỉ chạy trên một hệ điều hành nhất định thuộc một thiết bị nhất định, với ngôn ngữ phát triển cụ thể (như Objective-C hoặc Swift cho hệ điều hành iOS hoặc Java cho hệ điều hành Android), và thường phải thông qua điều chỉnh để có thể chạy được trên nhiều thiết bị khác nhau ^[4].

Vì ứng dụng được phát triển hoàn toàn trong môi trường dành riêng cho một hệ điều hành nhất định, với những đặc tính kỹ thuật và giao diện đặc trưng của hệ điều hành đó, nên không chỉ có lợi thế về hiệu suất, ứng dụng di động thuần túy còn có lợi thế trong trải nghiệm người dùng. Lợi thế trong trải nghiệm người dùng ở đây là ứng dụng có được sự đồng nhất về mặt giao diện và cảm nhận với nhiều ứng dụng thuần túy khác trên thiết bị. Người dùng có thể dễ dàng nắm bắt cách thức sử dụng cũng như tương tác với ứng dụng một cách nhanh chóng hơn. Ngoài ra, những ứng dụng di động thuần túy có lợi thế không nhỏ trong việc có khả năng truy cập và sử dụng một cách dễ dàng những tính năng đặc thù của thiết bị (GPS, sổ địa chỉ, camera, bộ phận cảm ứng...) ^[3].

Phát triển ứng dụng di động lai (Hybrid Mobile Application Development)

Phát triển ứng dụng di động lai là phát triển ứng dụng dựa trên nền tảng Web, sử dụng công nghệ phổ biến là HTML5 và JavaScript, được đóng gói lại trong một thành phần thuần túy (native container). Thành phần này thực hiện việc tải phần lớn thông tin lên giao diện khi người dùng truy cập qua từng chức năng của ứng dụng ^[3].

So sánh	Phát triển ứng dụng di động thuần túy	Phát triển ứng dụng di động lai
Điểm mạnh	<ul style="list-style-type: none">➤ Về mặt hiệu năng, trong hầu hết trường hợp, ứng dụng di động thuần túy chạy nhanh hơn ứng dụng lai nhưng sự chênh lệch không quá lớn và thường khó nhận biết bởi người dùng.➤ Về tính năng, ứng dụng di động thuần túy có thể dễ dàng truy cập tới phần cứng của thiết bị (camera, thiết bị thu âm...) cũng như truy cập tới các chức năng đặc quyền như sao chép, tạo, ghi dữ liệu trên bộ nhớ, thông tin danh bạ, cuộc gọi, tin nhắn... so với ứng dụng di động lai còn nhiều hạn chế về việc tận dụng toàn bộ sức mạnh của thiết bị.➤ Khi không có kết nối Internet, ứng dụng di động thuần túy có thể sử dụng những dữ liệu đã lưu trữ tạm (cache)	<ul style="list-style-type: none">➤ Người phát triển không bị hạn chế vào một hệ điều hành nhất định, có thể phát triển chỉ một ứng dụng nhưng chạy được trên nhiều hệ điều hành khác nhau.➤ Chi phí cho việc phát triển, bảo trì và nâng cấp ứng dụng sẽ được giảm thiểu đáng kể vì chỉ có một phiên bản duy nhất.➤ Ngôn ngữ lập trình cho ứng dụng di động lai là HTML và JavaScript rất phổ dụng, đa số người phát triển đều biết.

	trước đó trong khi ứng dụng lai đa phần đều cần phải có kết nối Internet để truy cập dữ liệu.	
Điểm yếu	<ul style="list-style-type: none"> ➤ Ứng dụng di động thuần túy không thể chạy trên nhiều hệ điều hành khác nhau, nói cách khác, một ứng dụng di động thuần túy chỉ chạy được trên một hệ điều hành nhất định. ➤ Khi muốn phát triển đa nền tảng, phát triển ứng dụng di động thuần túy sẽ có chi phí phát triển cao vì đòi hỏi khả năng thành thạo nhiều ngôn ngữ của người phát triển. ➤ Sử dụng đa dạng các phiên bản hệ điều hành gây cản trở tính tương thích của các ứng dụng di động thuần túy. 	<ul style="list-style-type: none"> ➤ Về mặt hiệu năng, đa phần các ứng dụng di động lai không chạy nhanh bằng các ứng dụng di động thuần túy. ➤ Ứng dụng di động lai thường không tận dụng được tối đa các tính năng phần cứng, sức mạnh của thiết bị. ➤ Một số chợ ứng dụng sẽ không chấp nhận ứng dụng lai được đăng lên nếu như không hoạt động đủ trơn tru.

Bảng 6.1: So sánh điểm mạnh, điểm yếu của Phát triển ứng dụng di động thuần túy (Native Mobile Application Development) và Phát triển ứng dụng di động lai (Hybrid Mobile Application Development)^[4]

Để có thể hỗ trợ được nhiều nền tảng di động một cách dễ dàng và tiết kiệm thời gian, đồng thời vì kiến trúc ứng dụng không quá phức tạp và không đòi hỏi nhiều tới những tính năng phần cứng đặc thù của thiết bị, chúng tôi quyết định sẽ phát triển ứng dụng di động theo hướng **Phát triển ứng dụng di động lai (Hybrid Mobile Application Development)**.

6.3.2 Các công nghệ sử dụng

Theo hướng phát triển ứng dụng di động lai, ứng dụng cần được xây dựng trên nền tảng Web. Giao diện Web của ứng dụng sẽ được thiết kế tập trung vào việc hỗ trợ cho thiết bị di động. Sau đó, ứng dụng sẽ được chuyển tiếp từ nền tảng Web sang các nền tảng di động thông qua các công nghệ hỗ trợ.

Các công nghệ chính được sử dụng để phát triển ứng dụng:

- AngularJS – Công nghệ được sử dụng để phát triển ứng dụng trên nền tảng Web
- Apache Cordova – Công nghệ được sử dụng để chuyển tiếp ứng dụng từ nền tảng Web sang các nền tảng di động khác nhau.

AngularJS

AngularJS là một framework mạnh mẽ mã nguồn mở có cấu trúc hỗ trợ phát triển ứng dụng Web động. Framework này cho phép sử dụng HTML như là một ngôn ngữ mẫu, đồng thời cho phép mở rộng các cú pháp HTML để diễn đạt các thành phần của ứng dụng một cách rõ ràng và ngắn gọn. Hai tính năng chính

của AngularJS là Liên kết dữ liệu (Data Binding) và Tiêm nhiễm phụ thuộc (Dependency Injection) giúp loại bỏ phần lớn mã code mà người phát triển ứng dụng thường phải viết.

Các đặc tính của AngularJS:

- AngularJS hỗ trợ người phát triển viết ứng dụng theo mô hình MVC (Model View Controller).
- Các ứng dụng AngularJS có khả năng tương thích với hầu hết các trình duyệt web với nhiều phiên bản trên các nền tảng khác nhau.
- AngularJS là framework mã nguồn mở, hoàn toàn miễn phí và được sử dụng rộng rãi bởi hàng ngàn lập trình viên trên thế giới.

Các tính năng và thành phần cốt lõi của AngularJS:

- **Khung nhìn (View):** là những gì mà người dùng nhìn thấy được.
- **Mô hình (Model):** dữ liệu nằm trên View mà có thể tương tác.
- **Liên kết dữ liệu (Data Binding):** đồng bộ dữ liệu giữa 2 thành phần model và view.
- **Chỉ thị (Directive):** mở rộng các thẻ HTML với các đặc tính và yếu tố tự tạo.
- **Bộ quản lý (Controller):** xử lý các thao tác nghiệp vụ bên dưới các Khung nhìn.
- **Biểu thức (Expression):** truy cập các biến và hàm từ Phạm vi.
- **Phạm vi (Scope):** phạm vi nơi các Mô hình được lưu trữ để các Bộ quản lý, Chỉ thị và Biểu thức có thể truy cập.
- **Dịch vụ (Service):** những thao tác nghiệp vụ có thể sử dụng lại độc lập với các Khung nhìn.
- **Tiêm nhiễm phụ thuộc (Dependency Injection):** tạo và liên kết các đối tượng và hàm.

Với các tính năng hỗ trợ phù hợp với kiến trúc của ứng dụng, cộng với sự phổ biến và được sử dụng rộng rãi của AngularJS. Nhóm chúng tôi quyết định sử dụng công nghệ này để phát triển ứng dụng trên nền tảng Web.

Apache Cordova

Apache Cordova là một framework mã nguồn mở hỗ trợ phát triển ứng dụng di động trên nền tảng Web. Framework này cho phép sử dụng những kỹ thuật Web chuẩn như HTML5, CSS3 và JavaScript để phát triển. Ứng dụng Web được xây dựng sẽ được chuyển tiếp thành ứng dụng trên các nền tảng di động gốc và dựa trên các API ràng buộc chuẩn để truy cập tới các chức năng của từng thiết bị như cảm biến, dữ liệu, tình trạng mạng...

Apache Cordova thường được sử dụng khi:

- Người phát triển ứng dụng di động muốn mở rộng ứng dụng từ một nền tảng sang nhiều nền tảng khác, mà không cần phải phát triển lại toàn bộ ứng dụng theo từng ngôn ngữ lập trình và công cụ riêng của từng nền tảng.
- Người phát triển ứng dụng Web muốn đưa ứng dụng lên nhiều nền tảng cũng như nhiều cửa hàng ứng dụng khác nhau.
- Người phát triển ứng dụng di động muốn pha trộn nhiều thành phần thuộc nền tảng di động gốc và các thành phần này có thể truy cập tới các API cấp thiết bị.

Các nền tảng hỗ trợ:

	android	blackberry10	ios	Ubuntu	wp8 (Windows Phone 8)	windows (8.1, 10, Phone 8.1)	OS X
cordova CLI	✓ Mac, Windows, Linux	✓ Mac, Windows, Linux	✓ Mac	✓ Ubuntu	✓ Windows	✓	✓ Mac
Embedded WebView	✓	X	✓	✓	X	X	✓
Plugin Interface	✓	✓	✓	✓	✓	✓	✓

Hình 6.1: Bảng các nền tảng Apache Cordova hỗ trợ

Với tính đơn giản dễ sử dụng, đồng thời có khả năng hỗ trợ nhiều nền tảng di động khác nhau, đặc biệt là 3 nền tảng di động chính được sử dụng rộng rãi hiện nay là Android, iOS và Windows Phone, nhóm chúng tôi quyết định sử dụng công nghệ Apache Cordova để chuyển tiếp ứng dụng từ nền tảng Web sang các nền tảng di động khác nhau.

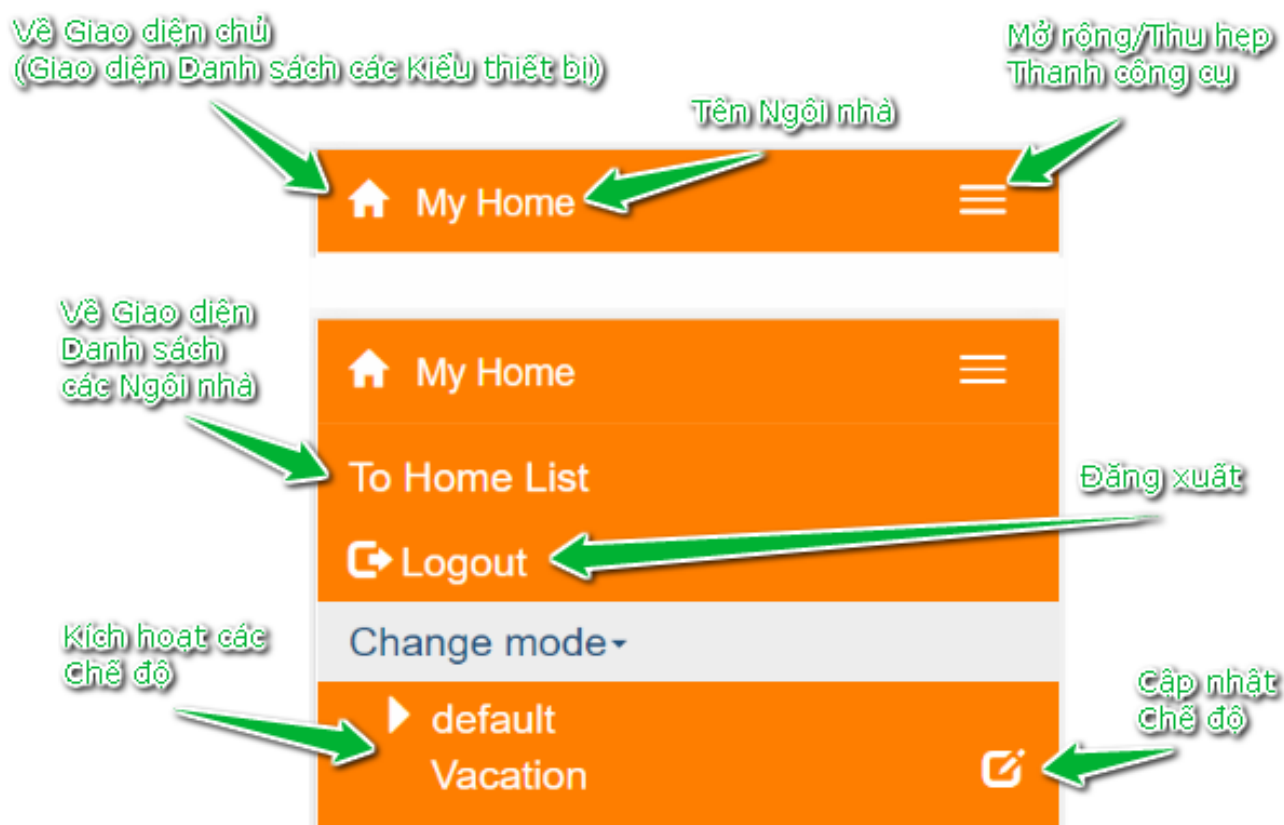
6.3.3 Hiện thực ứng dụng di động

Cấu trúc của ứng dụng di động bao gồm 3 mảng: **Các thành phần chính (components)**, **các thành phần đóng góp (shared)**, và **Dịch vụ chính (Main Service)**. Các thành phần chính phản ánh kiến trúc của ứng dụng di động ở Phần 5, với mỗi thành phần chính tương ứng với từng Giao diện chính của ứng dụng. Trong khi đó, các thành phần đóng góp là các thành phần chung được sử dụng lại nhiều lần giữa các thành phần chính khác nhau hoặc trong cùng một thành phần chính. Dịch vụ chính đảm nhiệm vai trò gọi các API Web Service từ phía Server để lấy dữ liệu, đồng thời thực hiện việc cập nhật trạng thái và đồng bộ liên kết giữa các thành phần chính và các thành phần đóng góp với nhau.

Các thành phần đóng góp (shared)

Các thành phần đóng góp trong ứng dụng bao gồm 8 thành phần: **Thanh công cụ (Navbar)**, **Thẻ nhà (Home-panel)**, **Thẻ kiểu thiết bị (Device-type-panel)**, **Thẻ thiết bị (Device-panel)**, **Thẻ kịch bản (Device-script-panel)**, **Thẻ kịch bản khi/thì (Device-script-when-then)**, **Thẻ kịch bản từ/đến (Device-script-from-to)** và **Thẻ kịch bản tự tạo (Device-script-custom)**. Các thành phần này thực chất là các **Chỉ thị (Directive)** của AngularJS, với mỗi thành phần có một giao diện với một bộ quản lý (controller) đảm nhận các xử lý nghiệp vụ riêng. Vì bản chất là Chỉ thị (Directive), mỗi thành phần đều có thể dễ dàng gắn vào nhau hoặc vào các thành phần chính như là một dạng thẻ HTML bình thường.

Thanh công cụ (Navbar)

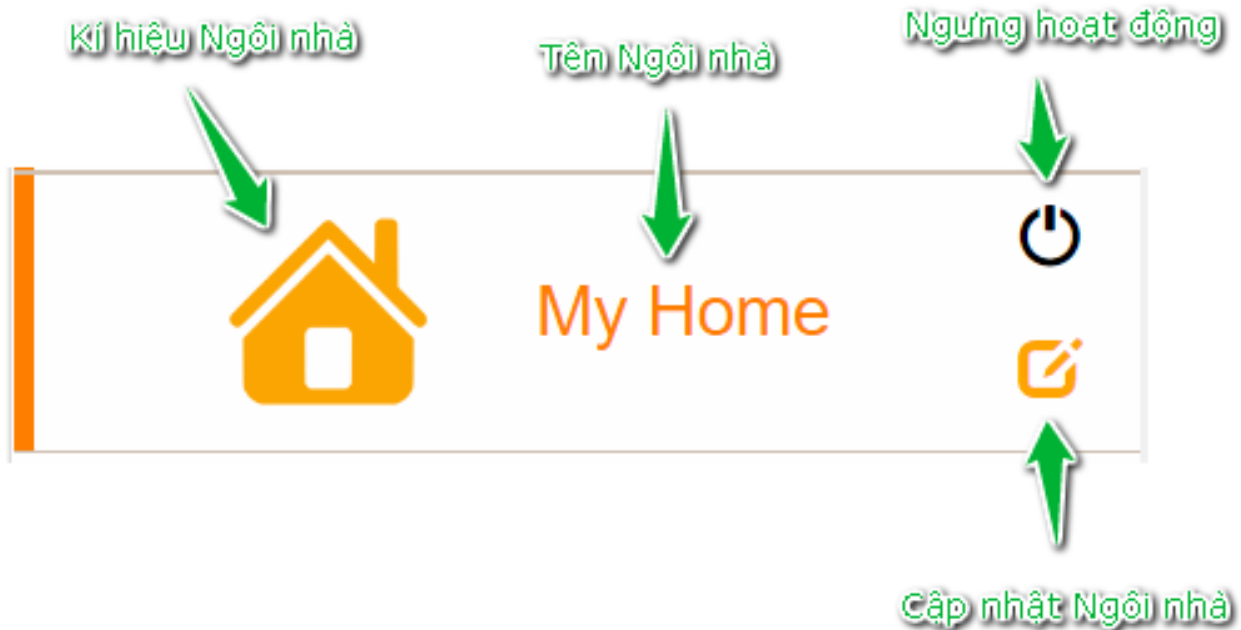


Hình 6.2: Thanh công cụ (Navbar)

Thanh công cụ (Navbar) là thành phần đóng góp cho phép người dùng thực hiện một số các thao tác cơ bản của ứng dụng một cách nhanh chóng, được sử dụng ở các thành phần chính là: Thành phần Danh sách các Ngôi nhà, Thành phần Danh sách các Kiểu thiết bị và Thành phần Danh sách các thiết bị. Thanh công cụ có nhiều chức năng khác nhau tùy thuộc vào thành phần chính mà nó được sử dụng. Các chức năng của thanh công cụ:

- Thể hiện tên của Ngôi nhà hiện tại đang truy cập.
- Cho phép người dùng quay lại Giao diện Danh sách các Kiểu thiết bị.
- Cho phép người dùng đăng xuất.
- Cho phép người dùng kích hoạt các Chế độ.
- Cho phép người dùng xem, cập nhật chỉnh sửa thông tin hoặc xóa các Chế độ.

Thẻ nhà (Home-panel)



Hình 6.3: Thẻ nhà





Thẻ nhà (Home-panel) là thành phần đóng góp thể hiện thông tin của một ngôi nhà được lắp đặt hệ thống, thành phần này được sử dụng nhiều lần như một danh sách trong thành phần chính Danh sách các Ngôi nhà. Các chức năng của thẻ nhà:

- Thể hiện tên của ngôi nhà.
- Cho phép người dùng ngưng hoạt động tạm thời (Disable) hoặc tái hoạt động (Enable) ngôi nhà.
- Cho phép người dùng xem, cập nhật chỉnh sửa thông tin hoặc xóa ngôi nhà.
- Chuyển tiếp người dùng tới thành phần chính Danh sách các Kiểu thiết bị.

Thẻ Kiểu thiết bị (Device-type-panel)

Kí hiệu Kiểu thiết bị

Tên Kiểu thiết bị

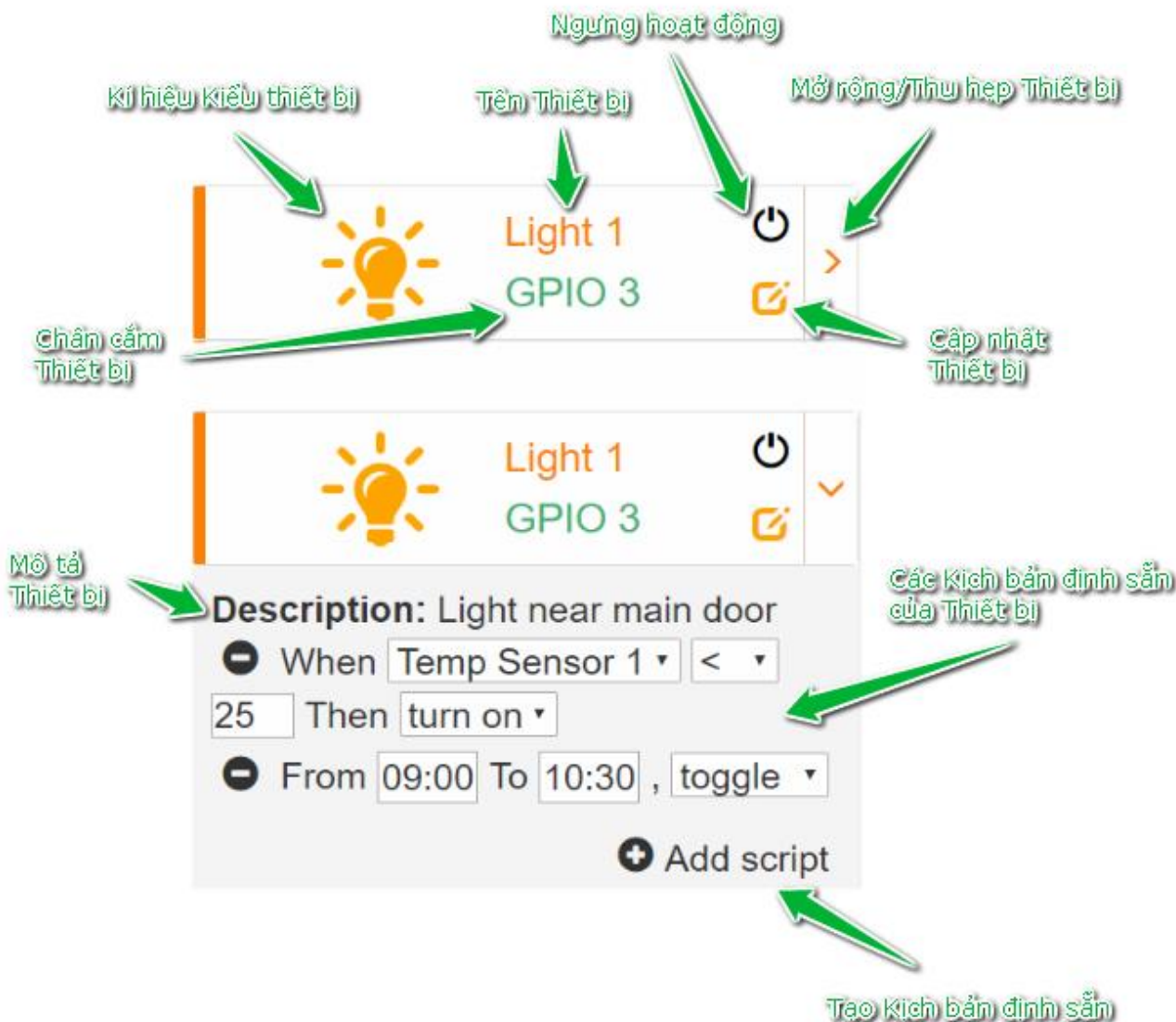
	Light Sensor
	Gas Sensor
	Light
	Buzzer

Hình 6.4: Một số Thẻ Kiểu thiết bị

Thẻ Kiểu thiết bị (Device-type-panel) là thành phần đóng góp thể hiện thông tin của một kiểu thiết bị được hệ thống hỗ trợ, thành phần này được sử dụng nhiều lần như một danh sách trong thành phần chính Danh sách Các kiểu thiết bị. Ứng với mỗi kiểu thiết bị thì thẻ Kiểu thiết bị có kí hiệu thể hiện và tên khác nhau, có 6 kiểu thiết bị hỗ trợ: Cảm biến nhiệt độ (Temperature Sensor), Cảm biến chuyển động (Motion Sensor), Cảm biến ánh sáng (Light Sensor), Cảm biến khí gas (Gas Sensor), Bóng đèn (Light) và Còi hú (Buzzer). Các chức năng của Thẻ Kiểu thiết bị:

- Thể hiện thông tin của Kiểu thiết bị.
- Chuyển tiếp người dùng tới thành phần chính Danh sách các thiết bị.

Thẻ Thiết bị (Device-panel)



Hình 6.5: Thẻ Thiết bị

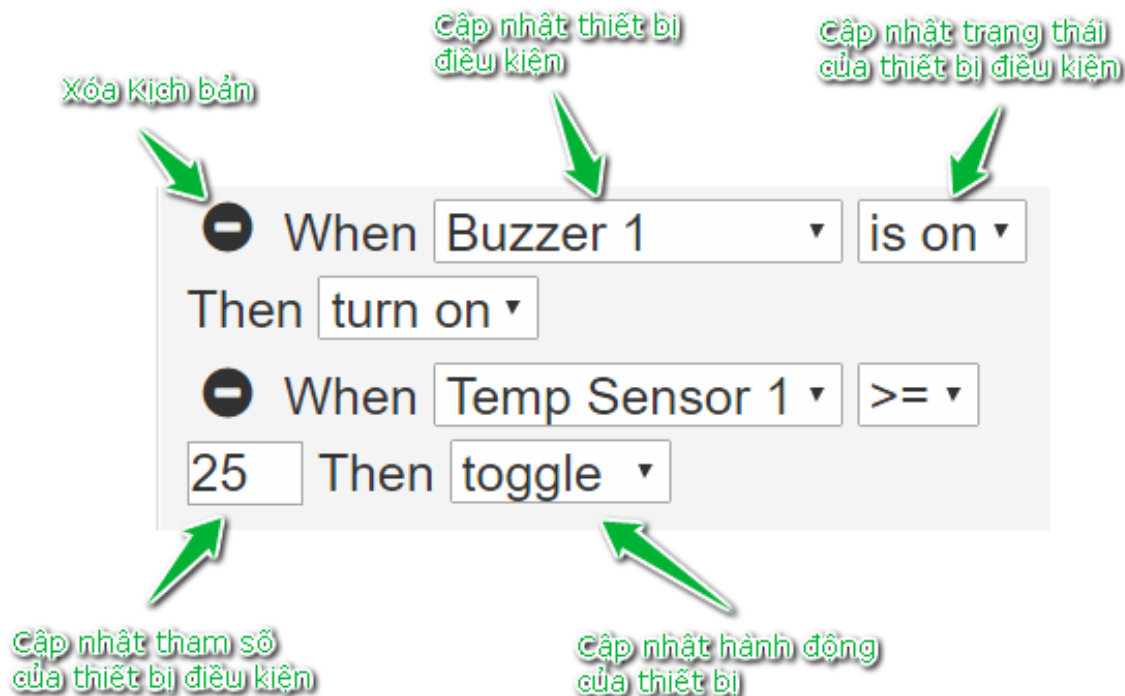
Thẻ Thiết bị (Device-panel) là thành phần đóng góp thể hiện thông tin của một thiết bị cùng với các Kịch bản định sẵn của thiết bị đó, được sử dụng nhiều lần như một danh sách trong thành phần chính Danh sách các Thiết bị. Thành phần này chứa bên trong nó một tập hợp các thành phần đóng góp Thẻ Kịch bản để hỗ trợ việc thể hiện thông tin các Kịch bản thuộc thiết bị. Các chức năng của Thẻ thiết bị:

- Thể hiện kiểu, tên và chân cắm của thiết bị.
- Cho phép người dùng ngưng hoạt động tạm thời (Disable) hoặc tái hoạt động (Enable) thiết bị.
- Cho phép người dùng xem, cập nhật chỉnh sửa thông tin hoặc xóa thiết bị.
- Cho phép người dùng xem thông tin các Kịch bản định sẵn của thiết bị.
- Cho phép người dùng tạo thêm Kịch bản định sẵn cho thiết bị.

Thẻ Kịch bản (Device-script-panel)

Thẻ Kịch bản (Device-script-panel) chứa bên trong nó 2 thành phần đóng góp khác là Thẻ Kịch bản khi/thì (Device-script-when-then) và Thẻ Kịch bản từ/đến (Device-script-from-to). Tùy vào loại Kịch bản được truyền vào mà thành phần này sẽ quyết định lựa chọn một trong hai thành phần đóng góp bên trong nó để thể hiện.

Thẻ Kịch bản khi/thì (Device-script-when-then)



Hình 6.6: Một số Thẻ Kịch bản khi/thì

Thẻ Kịch bản khi/thì (Device-script-when-then) là thành phần đóng góp thể hiện thông tin của một kịch bản thuộc loại Khi/thì (When/Then), với 2 bộ phận là Điều kiện và Hành động. Bộ phận Điều kiện gồm 3 bộ phận nhỏ hơn là Thiết bị điều kiện, Trạng thái của thiết bị điều kiện và Tham số của thiết bị điều kiện. Tham số của thiết bị điều kiện có được thể hiện hay không tùy thuộc vào kiểu của thiết bị điều kiện. Các chức năng của Thẻ Kịch bản khi/thì:

- Thể hiện nội dung bao gồm Điều kiện và Hành động của kịch bản thuộc loại Khi/thì.
- Cho phép người dùng cập nhật chỉnh sửa thiết bị điều kiện.
- Cho phép người dùng cập nhật chỉnh sửa trạng thái của thiết bị điều kiện.
- Cho phép người dùng cập nhật chỉnh sửa tham số của thiết bị điều kiện.
- Cho phép người dùng cập nhật chỉnh sửa hành động của thiết bị.
- Cho phép người dùng xóa kịch bản.

Thẻ Kịch bản từ/đến (Device-script-from-to)



Hình 6.7: Thẻ Kịch bản từ/đến

Thẻ Kịch bản từ/đến (Device-script-from-to) là thành phần đóng góp thể hiện thông tin của một kịch bản thuộc loại Từ/đến (From/To), với 2 bộ phận là Khoảng thời gian hành động được thực hiện và Hành động. Các chức năng của thẻ Kịch bản từ/đến:

- Thể hiện nội dung bao gồm Khoảng thời gian hành động được thực hiện và Hành động của kịch bản thuộc loại từ/đến.
- Cho phép người dùng cập nhật chỉnh sửa khoảng thời gian thực hiện hành động của thiết bị.
- Cho phép người dùng cập nhật chỉnh sửa hành động của thiết bị.
- Cho phép người dùng xóa kịch bản.