|  |
| --- |
| ĐẠI HỌC BÁCH KHOA HÀ NỘI  **VIỆN ĐIỆN TỬ - VIỄN THÔNG**  logo_128  ĐỒ ÁN  **TỐT NGHIỆP ĐẠI HỌC**  **Đề tài:**  **XÂY DỰNG ỨNG DỤNG**  **CẢNH BÁO CHÁY**  Sinh viên thực hiện: LÊ QUANG HUY  Lớp ĐTVT.11-K62  Giảng viên hướng dẫn: PGS NGUYỄN XUÂN QUYỀN  Hà Nội, 02-2024 |

|  |
| --- |
| ĐẠI HỌC BÁCH KHOA HÀ NỘI  **VIỆN ĐIỆN TỬ - VIỄN THÔNG**  logo_128  ĐỒ ÁN  **TỐT NGHIỆP ĐẠI HỌC**  **Đề tài:**  **XÂY DỰNG ỨNG DỤNG**  **CẢNH BÁO CHÁY**  Sinh viên thực hiện: LÊ QUANG HUY  Lớp ĐTVT.11-K62  Giảng viên hướng dẫn: PGS NGUYỄN XUÂN QUYỀN  Cán bộ phản biện:  Hà Nội, 02-2024 |

**ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP**

(Dùng cho giảng viên hướng dẫn)

Tên giảng viên đánh giá: PGS Nguyễn Xuân Quyền

Họ và tên sinh viên: Lê Quang Huy MSSV: 20172605

Tên đồ án: Xây dựng hệ thống cảnh báo cháy

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**

Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thực nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến; căn lề thống nhất, có dấu cách sau dấu chấm, dấu phảy v.v.), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5)** *(chọn 1 trong 3 trường hợp)* | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/Đạt giải SVNCKH giải 3 cấp Viện trở lên/Có giải thưởng khoa học (quốc tế hoặc trong nước) từ giải 3 trở lên/Có đăng ký bằng phát minh, sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị SVNCKH nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành (VD: TI contest) | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | |  | | | | |

***Nhận xét khác*** *(về thái độ và tinh thần làm việc của sinh viên)*

|  |  |
| --- | --- |
|  | Ngày: … / … / 20…  **Người nhận xét**  (Ký và ghi rõ họ tên)  Nguyễn Xuân Quyền |

**ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP**

(Dùng cho cán bộ phản biện)

Giảng viên đánh giá:

Họ và tên sinh viên: Lê Quang Huy MSSV: 20172605

Tên đồ án: Xây dựng hệ thống cảnh báo cháy

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**

Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thực nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến; căn lề thống nhất, có dấu cách sau dấu chấm, dấu phảy v.v.), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5)** *(chọn 1 trong 3 trường hợp)* | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/Đạt giải SVNCKH giải 3 cấp Viện trở lên/Có giải thưởng khoa học (quốc tế hoặc trong nước) từ giải 3 trở lên/Có đăng ký bằng phát minh, sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị SVNCKH nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành (VD: TI contest) | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | |  | | | | |

***Nhận xét khác của cán bộ phản biện***

|  |  |
| --- | --- |
|  | Ngày: … / … / 20…  **Người nhận xét**  (Ký và ghi rõ họ tên) |

**LỜI NÓI ĐẦU**

Ngành Công nghệ thông tin và hệ thống IoT – Internet of Things là vấn đề đang rất được quan tâm, chú trọng hiện nay và có thể phát triển rộng rãi hơn trong tương lai. Do tính tiện dụng nên hệ thống IoT được coi là xu hướng phát triển nổi bật trong thời kỳ công nghiệp 4.0. Đây là thời kỳ của trí tuệ nhân tạo, tự động hóa với các thiết bị thông minh được kết nối với nhau thông qua mạng internet (vạn vật kết nối). Các thiết bị trong hệ thống đều được lập trình để có thể làm việc tự động, giúp cuộc sống của con người hiện đại hơn, thông minh hơn, có thể tiết kiệm thời gian và chi phí.

Hàng năm, trên toàn quốc xảy ra hang ngàn vụ cháy, gây thiệt hại to lớn về người và tài sản. Trong năm 2023, thông tin từ Cục Cảnh sát PCCC&CNCH, đã có tổng cộng 3440 vụ cháy, làm chết 146 người, bị thương 109 người, thiệt hại về tài sản sơ bộ ước tính 878 tỷ đồng và 236 ha rừng. Trong đó, số vụ cháy nhà dân vẫn chiếm tỷ lệ cao nhất với 1016 vụ cháy (29.5%). Vì vậy, việc lắp đặt các hệ thống cảnh báo cháy và chữa cháy có vai trò rất quan trọng. Nhận ra những thiếu xót và bất cập như trên, hệ thống “Cảnh báo cháy” được đưa ra phân tích và thiết kế, với mong muốn giúp ngăn chặn và xử lý kịp thời các đám cháy khi con người chưa thể can thiệp được.

Trước nhu cầu của người dùng và hướng phát triển trên, tác giả đã quyết định chọn đề tài: “XÂY DỰNG ỨNG DỤNG CẢNH BÁO CHÁY” với mục đích nghiên cứu, tìm hiểu và xây dựng ứng dụng có đủ những chức năng cần thiết. Ứng dụng này cho phép theo dõi các chỉ số trong các căn hộ, tòa nhà thông qua các cảm biến, từ đó sẽ có hướng xử lý để cảnh báo và chữa cháy hợp lý. Nội dung đồ án gồm có 3 chương:

*Chương 1: Tổng quan ứng dụng cảnh báo cháy*. Chương này sẽ trình bày về mô hình tổng quan, mô tả rõ hoạt động của hệ thống, giới thiệu một số giải pháp và công cụ được sử dụng trong đề tài.

*Chương 2: Phân tích thiết kế hệ thống*. Chương này trình bày những yêu cầu của hệ thống về chức năng và phi chức năng. Ngoài ra, còn tạo ra sơ đồ Use Case, tìm hiểu thiết kế biểu đồ lớp (Class Diagram) cho Models chung và cuối cùng là thiết kế luồng đi của hệ thống, luồng đi của dữ liệu thông qua biểu đồ hoạt động (Activity Diagram) và biểu đồ tuần tự (Sequence Diagram).

*Chương 3: Triển khai hệ thống*. Chương này trình bày về quá trình tiến hành triển khai ứng dụng cảnh báo cháy.

Do kiến thức vẫn còn hạn chế, bài báo cáo này không tránh khỏi những sai sót. Tác giả rất mong nhận được những lời góp ý từ quý thầy cô để đồ án tốt nghiệp được hoàn thiện và giúp tác giả có thêm những kinh nghiệm quý báu.

**LỜI CẢM ƠN**

Trong suốt khoảng thời gian học tập tại Bách Khoa, tác giả đã có nhiều kỷ niệm vui buồn, và trải qua quá trình học tập kiến thức cũng như rèn luyện kỹ năng mềm cho bản thân. Tác giả xin dành lời cảm ơn sâu sắc đến các thầy cô đã tận tình dạy dỗ, truyền đạt kiến thức, kinh nghiệm và hỗ trợ sinh viên hết sức mình trong suốt những năm vừa qua.

Tác giả vô cùng biết ơn giảng viên hướng dẫn PGS Nguyễn Xuân Quyền đã tận tình hướng dẫn, chỉ bảo, góp ý trong suốt quá trình thực hiện đề tài để tác giả có thể hoàn thành đồ án của mình. Mặc dù tình hình dịch căng thẳng, khó khăn khiến quá trình làm việc phải thực hiện online nhưng thầy đã chỉ dạy, giúp đỡ sinh viên hết sức tận tình khi gặp khó khăn trong lúc thực hiện đồ án. Tác giả cũng vô cùng biết ơn gia đình và bạn bè, những người đã luôn luôn ở bên cạnh, động viên, chia sẻ trong suốt thời gian thực hiện đề tài đồ án tốt nghiệp.

Cuối cùng, tác giả xin kính chúc các thầy cô Đại học Bách khoa Hà Nội nói chung, các thầy cô trường Điện – Điện tử Viện Điện tử - Viễn thông nói riêng sức khỏe dồi dào, thành công trong sự nghiệp giảng dạy cao quý của mình.

**LỜI CAM ĐOAN**

Tôi là Lê Quang Huy, mã số sinh viên là 20172605, sinh viên lớp ĐTVT-11-K62, khóa 62. Người hướng dẫn là PGS Nguyễn Xuân Quyền. Tôi xin cam đoan toàn bộ nội dung được trình bày trong đồ án “*Xây dựng ứng dụng cảnh báo cháy*” là kết quả quá trình tìm hiểu và nghiên cứu của tôi. Mọi thông tin trích dẫn đều tuân thủ các quy định về sở hữu trí tuệ, các tài liệu tham khảo được liệt kê rõ ràng. Tôi xin chịu hoàn toàn trách nhiệm với những nội dung được viết trong đồ án này.

Hà Nội, ngày ... tháng ... năm 2021

Người cam đoan

**Lê Quang Huy**

**MỤC LỤC**

[DANH MỤC HÌNH VẼ iii](#_Toc96992117)

[DANH MỤC BẢNG BIỂU v](#_Toc96992118)

[DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT vi](#_Toc96992119)

[TÓM TẮT ĐỒ ÁN vii](#_Toc96992120)

[CHƯƠNG 1. TỔNG QUAN HỆ THỐNG THEO DÕI BỆNH NHÂN TỪ XA 1](#_Toc96992121)

[1.1 Lý do chọn đề tài 1](#_Toc96992122)

[1.2 Giới thiệu tổng quan hệ thống 2](#_Toc96992123)

[1.2.1 Mô hình hệ thống 2](#_Toc96992124)

[1.2.2 Nguyên tắc hoạt động của hệ thống 3](#_Toc96992125)

[1.3 Phương thức kết nối 3](#_Toc96992126)

[1.3.1 Đánh giá khả năng kết nối 3](#_Toc96992127)

[1.3.2 Tổng quan về MQTT 4](#_Toc96992128)

[1.4 Cơ sở dữ liệu 5](#_Toc96992129)

[1.4.1 So sánh cơ sở dữ liệu SQL và NoSQL 5](#_Toc96992130)

[1.4.2 Tổng quan về MongoDB 6](#_Toc96992131)

[1.5 Tổng quan về mô hình Model-View-Controller (MVC) 6](#_Toc96992132)

[1.5.1 Khái niệm 6](#_Toc96992133)

[1.5.2 Đặc điểm của mô hình MVC 7](#_Toc96992134)

[1.6 Giới thiệu về công cụ và kỹ thuật được sử dụng 7](#_Toc96992135)

[1.6.1 Ngôn ngữ C# 7](#_Toc96992136)

[1.6.2 Visual Studio 8](#_Toc96992137)

[1.6.3 Xamarin 10](#_Toc96992138)

[1.6.4 Windows Presentation Foundation (WPF) 11](#_Toc96992139)

[1.7 Kết luận 12](#_Toc96992140)

[CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 13](#_Toc96992141)

[2.1 Yêu cầu hệ thống 13](#_Toc96992142)

[2.1.1 Yêu cầu chức năng 13](#_Toc96992143)

[2.1.2 Yêu cầu phi chức năng 13](#_Toc96992144)

[2.2 Xây dựng Use case Diagram 13](#_Toc96992145)

[2.2.1 Tìm hiểu biểu đồ Use case 13](#_Toc96992146)

[2.2.2 Bản vẽ Use case 15](#_Toc96992147)

[2.3 Xây dựng Class Diagram 19](#_Toc96992148)

[2.3.1 Tìm hiểu biểu đồ lớp (Class Diagram) 19](#_Toc96992149)

[2.3.2 Bản vẽ Class Diagram 21](#_Toc96992150)

[2.4 Xây dựng Activity Diagram 22](#_Toc96992151)

[2.4.1 Tìm hiểu biểu đồ hoạt động (Activity Diagram) 22](#_Toc96992152)

[2.4.2 Bản vẽ Activity Diagram 24](#_Toc96992153)

[2.5 Xây dựng Sequence Diagram 30](#_Toc96992154)

[2.5.1 Tìm hiểu biểu đồ tuần tự (Sequence Diagram) 30](#_Toc96992155)

[2.5.2 Bản vẽ Sequence Diagram 31](#_Toc96992156)

[2.6 Xây dựng giao thức liên lạc 33](#_Toc96992157)

[2.7 Kết luận 35](#_Toc96992158)

[CHƯƠNG 3. TRIỂN KHAI MÔ HÌNH MVC TRÊN XAMARIN 36](#_Toc96992159)

[3.1 Tìm hiểu về Xamarin và Xamarin Forms 36](#_Toc96992160)

[3.1.1 Xamarin 36](#_Toc96992161)

[3.1.2 Xamarin Forms 38](#_Toc96992162)

[3.2 Triển khai trên Xamarin 41](#_Toc96992163)

[3.2.1 Tạo project trên Xamarin (Windows) 41](#_Toc96992164)

[3.2.2 Tiến hành Code phần Base 43](#_Toc96992165)

[3.3 Giao diện của một số chức năng đã thực hiện 49](#_Toc96992166)

[3.3.1 Giao diện đăng nhập 49](#_Toc96992167)

[3.3.2 Giao diện danh sách bệnh nhân 50](#_Toc96992168)

[3.3.3 Giao diện cài đặt 51](#_Toc96992169)

[3.3.4 Giao diện đổi mật khẩu 52](#_Toc96992170)

[3.4 Kết luận 53](#_Toc96992171)

[KẾT LUẬN 54](#_Toc96992172)

[Kết luận chung 54](#_Toc96992173)

[Hướng phát triển 54](#_Toc96992174)

[TÀI LIỆU THAM KHẢO 55](#_Toc96992175)

DANH MỤC HÌNH VẼ

[Hình 1.1 Mô hình tổng quan của hệ thống 2](#_Toc157637438)

[Hình 1.2 Mô hình MVC 7](#_Toc157637439)

[Hình 2.1 Sơ đồ Use Case tổng quát của hệ thống 14](#_Toc157637440)

[Hình 2.2 Sơ đồ use case cho admin 15](#_Toc157637441)

[Hình 2.3 Sơ đồ use case cho người dùng 16](#_Toc157637442)

[Hình 2.4 Usecase quản lý căn hộ 16](#_Toc157637443)

[Hình 2.5 Usecase quản lý thiết bị 17](#_Toc157637444)

[Hình 2.6 Ví dụ về các tính chất cơ bản của class 17](#_Toc157637445)

[Hình 2.7 Ví dụ về các phạm vi truy cập của class 18](#_Toc157637446)

[Hình 2.8 Các loại quan hệ trong class diagram 18](#_Toc157637447)

[Hình 2.9 Bản vẽ class diagram cho Model 20](#_Toc157637448)

[Hình 2.10 Biểu đồ hoạt động chức năng đăng nhập 23](#_Toc157637449)

[Hình 2.11 Biểu đồ hoạt động chức năng xem chỉ số căn hộ 24](#_Toc157637450)

[Hình 2.12 Biểu đồ hoạt động chức năng cảnh báo 25](#_Toc157637451)

[Hình 2.13 Biểu đồ hoạt động chức năng quản lý căn hộ 26](#_Toc157637452)

[Hình 2.14 Biểu đồ hoạt động chức năng quản lý thiết bị 27](#_Toc157637453)

[Hình 2.15 Biểu đồ tuần tự chức năng đăng nhập 29](#_Toc157637454)

[Hình 2.16 Biểu đồ tuần tự chức năng xem chỉ số căn hộ 30](#_Toc157637455)

[Hình 2.17 Biểu đồ tuần tự chức năng cảnh báo 31](#_Toc157637456)

[Hình 3.1 Framework và reference của Web API Server 32](#_Toc157637457)

[Hình 3.2 Cấu trúc solution API Web Server 33](#_Toc157637458)

[Hình 3.3 Cấu trúc của web client 34](#_Toc157637459)

[Hình 3.4 Sơ đồ kết nối ESP8266 và cảm biến DHT11 38](#_Toc157637460)

[Hình 3.5 Cảm biến khí gas MQ6 39](#_Toc157637461)

[Hình 3.6 Module ESP32-CAM Wifi Bluetooth OV2640 kèm đế 45](#_Toc157637462)

[Hình 3.7 Giao diện đăng nhập 47](#_Toc157637463)

[Hình 3.8 Giao diện danh sách căn hộ 48](#_Toc157637464)

[Hình 3.9 Giao diện thêm mới căn hộ 48](#_Toc157637465)

[Hình 3.10 Giao diện chi tiết căn hộ và xem chỉ số căn hộ 49](#_Toc157637466)

[Hình 3.11 Giao diện danh sách thiết bị 49](#_Toc157637467)

[Hình 3.12 Giao diện thêm mới thiết bị 50](#_Toc157637468)

[Hình 3.13 Giao diện chi tiết thiết bị 50](#_Toc157637469)

[Hình 3.14 Triển khai lắp linh kiện trên board test 52](#_Toc157637470)

DANH MỤC BẢNG BIỂU

[Bảng 1.1 So sánh MQTT protocols và HTTP protocols 3](#_Toc157637471)

[Bảng 1.2 So sánh cơ sở dữ liệu SQL và NoSQL 5](#_Toc157637472)

[Bảng 2.1 Các phần tử của biểu đồ Use Case 12](#_Toc157637473)

[Bảng 2.2 Các thành phần của biểu đồ tuần tự 28](#_Toc157637474)

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| **Kí hiệu** | **Thuật ngữ đầy đủ** |
| IoT | Internet of Thing |
| MQTT | Message Queuing Telemetry Transport |
| MVC | Model-View-Controller |
| NoSQL | Non relational Structured Query Language |
| SQL | Structured Query Language |
| RDBMS | Relational Database Management System |
| HTTP | Hyper Text Transfer Protocol |
| XAML | eXtensible Application Markup Language |
| IDE | Integrated Developement Environment |

# TÓM TẮT ĐỒ ÁN

Hiện nay, với sự phát triển mạnh mẽ của Công nghiệp 4.0, các hệ thống IoT đang ngày càng phát triển để phục vụ đời sống con người hiện đại, thông minh hơn. Đến từ nhu cầu cảnh báo cháy và chữa cháy kịp thời, việc phát triển ứng dụng cảnh báo cháy là thực sự cần thiết.

Đề tài “Xây dựng ứng dụng cảnh báo cháy” có nội dung chính là phân tích thiết kế các chức năng các luồng dữ liệu của hệ thống, tìm hiểu về Web Api, giao thức MQTT, ReactJS – framework để xây dựng giao diện người dùng trên nền tảng web, nắm được tổng thể các phần trong một ứng dụng và cách thiết kế nên ứng dụng. Với mục đích trau dồi kỹ năng làm việc nhóm, phân tích thiết kế hệ thống và khả năng tự tìm hiểu, nâng cao kiến thức về các kỹ thuật công nghệ mới, tác giả đã thực hành và viết nên ứng dụng cảnh báo cháy này bằng phần mềm Microsoft Visual Studio, dựa trên ngôn ngữ C#. Đồ án này gồm có 3 chương, trong đó chương đầu tiên sẽ trình bày về mô hình tổng quan, mô tả rõ hoạt động của hệ thống, giới thiệu một số giải pháp và công cụ được sử dụng trong đề tài. Chương thứ hai trình bày những yêu cầu của hệ thống về chức năng và phi chức năng, thiết kế các biểu đồ chức năng, biểu đồ lớp, biểu đồ hoạt động và biểu đồ tuần tự. Chương cuối cùng trình bày về quá trình tiến hành triển khai ứng dụng cảnh báo cháy.

Trong gần 4 tháng thực hiện đề tài, ứng dụng đã đạt được một số chức năng cơ bản cho một ứng dụng cảnh báo cháy và một số giao diện cơ sở để ứng dụng có thể dễ dàng phát triển hơn trong tương lai.

**ABSTRACT**

Currently, with the strong development of Industry 4.0, IoT systems are increasingly developing to serve modern, smarter human life. Due to the need for timely fire warning and fire fighting, developing a fire warning application is really necessary.

The topic "Building a fire warning application" has the main content of analyzing and designing the functions of the system's data flows, learning about Web Api, MQTT protocol, ReactJS - framework to build user interfaces. used on the web platform, understanding the overall parts of an application and how to design the application. With the purpose of cultivating teamwork skills, system design analysis and the ability to self-study and improve knowledge of new technology techniques, the author has practiced and written this fire warning application. using Microsoft Visual Studio software, based on C# language. This project includes 3 chapters, of which the first chapter will present the overall model, clearly describe the operation of the system, and introduce some solutions and tools used in the project. The second chapter presents functional and non-functional system requirements, designs function diagrams, class diagrams, activity diagrams and sequence diagrams. The final chapter presents the process of implementing fire warning applications.

During nearly 4 months of implementing the project, the application has achieved some basic functions for a fire warning application and some basic interfaces so that the application can be more easily developed in the future.

# TỔNG QUAN ỨNG DỤNG CẢNH BÁO CHÁY

Sau khi cần phân tích, chúng ta đã có cho những lý do phù hợp để chọn đề tài thực hiện này. Từ đó, lập kế hoạch và đặt mục tiêu để hoàn thành dự án. Nội dung sau sẽ trình bày về mô hình tổng quan, mô tả rõ hoạt động của hệ thống, giới thiệu một số giải pháp và công cụ được sử dụng trong đề tài.

## Lý do chọn đề tài

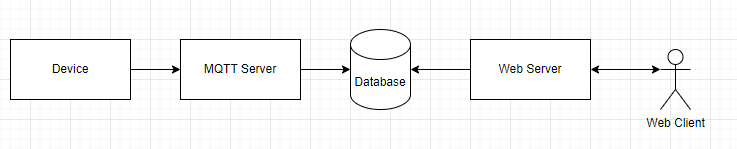
Thông tin từ Cục Cảnh sát PCCC&CNCH, trong năm 2023, toàn quốc xảy ra 3.440 vụ cháy, làm chết 146 người, bị thương 109 người, thiệt hại về tài sản sơ bộ ước tính khoảng 878 tỷ đồng. Ngoài ra còn ghi nhận 16 vụ nổ, làm 11 người chết, bị thương 27 người. Trong năm 2023, số vụ cháy nhà dân chiếm tỷ lệ cao nhất với 1.016 vụ cháy (chiếm 29,5%); các loại hình khác đều chiếm dưới 10%. Vì vậy, việc lắp đặt các hệ thống cảnh báo cháy và chữa cháy có vai trò rất quan trọng.

Trước nhu cầu của người dùng và hướng phát triển trên, tác giả đã quyết định chọn đề tài: “XÂY DỰNG ỨNG DỤNG CẢNH BÁO CHÁY” với mục đích nghiên cứu, tìm hiểu và xây dựng ứng dụng có đủ những chức năng cần thiết. Ứng dụng này cho phép theo dõi các chỉ số trong các căn hộ, tòa nhà thông qua các cảm biến, từ đó sẽ có hướng xử lý để cảnh báo và chữa cháy hợp lý.

Cụ thể, nếu áp dụng hệ thống này, người dùng có thể theo dõi các chỉ số như nhiệt độ, độ ẩm, khí gas trong căn hộ của mình. Nếu các chỉ số có bất thường, hệ thống sẽ tự động cảnh báo cho người dùng giúp người dùng có thể đưa ra những phương án xử lý kịp thời và hợp lý.

## Giới thiệu tổng quan hệ thống

### Mô hình hệ thống



Hình . Mô hình tổng quan của hệ thống

Mô tả hệ thống:

* Hệ thống gồm các khối device, MQTT server, Web server, client và database.
* Device bao gồm nhiều cảm biến đo các chỉ số, các chỉ số được cập nhật liên tục theo giời gian thực, mỗi khi phát hiện bất thường device sẽ có nhiệm vụ phát cảnh báo và báo về MQTT server.
* MQTT server nhận các bản tin và lưu các chỉ số nhận được vào database.
* Client có thể bao gồm các ứng dụng web, mobile và desktop được cung cấp cho các end-users để theo dõi thông tin và nhận cảnh báo mỗi khi xảy ra bất thường.
* Client sẽ giao tiếp với web server thông qua web api.

### Nguyên tắc hoạt động của hệ thống

Hệ thống được thiết lập để kết nối các thiết bị đo lường với các thiết bị điện tử như là smart phone, laptop, ... nhằm giúp cho việc theo dõi các chỉ số và đưa ra cảnh báo khi có nguy cơ cháy nổ một cách kịp thời, chuẩn xác.

Hệ thống cảnh báo cháy gồm các thiết bị điện tử để đo các chỉ số và liên tục gửi các chỉ số về MQTT server theo thời gian thực; một MQTT server để nhận và lưu thông tin các chỉ số nhận được từ các thiết bị điện tử vào database; một Web server được kết nối với các client (web app, mobile, desktop app) thông qua web api.

## Phương thức kết nối

### Đánh giá khả năng kết nối

Về phần giao thức kết nối, có khá nhiều loại trong đó có HTTP, MQTT, ... Trong đó giao thức MQTT là giao thức được lựa chọn để sử dụng trong hệ thống vì đây là một giao thức nhẹ, lý tưởng cho hầu hết các kết nối IoT và M2M vì yêu cầu băng thông thấp và có độ tin cậy cao.

Sự so sánh về đặc điểm của hai giao thức kết nối MQTT và HTTP được thể hiện rõ trong bảng 1-1 [1].

Bảng . So sánh MQTT protocols và HTTP protocols

|  |  |  |
| --- | --- | --- |
| Đặc điểm | MQTT PROTOCOL | HTTP PROTOCOL |
| Tên đầy đủ | Message Queuing Telemetry Transport | Hyper Text Transfer Protocol |
| Kích thước bản tin | nhỏ, là mã nhị phân với header là 2 byte | lớn, mã hóa bằng ASCII |
| Bảo mật dữ liệu | Có | Không, https mới cung cấp bảo mật dữ liệu |
| Độ phức tạp | đơn giản | phức tạp |

### Tổng quan về MQTT

#### Khái niệm

MQTT (Message Queuing Telemetry Transport) là một giao thức gửi dạng publish/subscribe sử dụng cho các thiết bị (IoT) với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Được thiết kế nhẹ và linh hoạt, do đó nó chỉ có một lớp bảo mật ở tầng ứng dụng: bảo mật bằng xác thực (xác thực các client được truy cập tới broker).

Bởi vì giao thức này sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M [2].

#### Đặc điểm

Trong một hệ thống sử dụng giao thức MQTT, nhiều client kết nối tới một server (trong MQTT, server được gọi là MQTT Broker). Mỗi client sẽ đăng kí theo dõi các kênh thông tin (gọi là Topic) hoặc gửi dữ liệu lên kênh thông tin đó. Quá trình đăng ký này gọi là “Subscribe” và hành động một client gửi dữ liệu lên kênh thông tin được gọi là “Publish”. Mỗi khi thông tin đó được cập nhật dữ liệu (dữ liệu này có thể đến từ các client khác) thì những client nào đăng ký theo dõi kênh này sẽ nhận được dữ liệu cập nhật đó.

#### Ưu điểm

MQTT có đặc trưng riêng và rất lý tưởng để sử dụng trong các môi trường bị giới hạn tài nguyên như:

* Những nơi mà giá mạng quá đắt hoặc băng thông thấp, hoặc độ tin cậy thấp
* Khi chạy trên một thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ
* Việc truyền message là luôn không quan tâm đến nội dung truyền

MQTT được thiết kế nhằm phục vụ truyền thông machine – to – machine nhưng thực tế chứng minh nó lại linh hoạt hơn mong đợi. Nó hoàn toàn có thể áp dụng cho các kịch bản truyền thông khác nhau như: machine - to – cloud, cloud – to – machine, app – to – app. Chỉ cần có một broker phù hợp và MQTT client được cài đặt đúng cách, các thiết bị xây dựng trên nhiều nền tảng khác nhau có thể giao tiếp với nhau một cách dễ dàng.

### Tổng quan về HTTP

#### Khái niệm

Http (HyperText Transfer Protocol) là giao thức truyền tải siêu văn bản được sử dụng trong www dùng để truyền tải dữ liệu giữa Web server đến các trình duyệt Web và ngược lại.

Http hoạt động dựa trên mô hình Client (máy khách) – Server (máy chủ). Các máy tính của người dùng sẽ đóng vai trò làm máy khách (Client). Sau một thao tác nào đó của người dùng, các máy khách sẽ gửi yêu cầu đến máy chủ (Server) và chờ đợi câu trả lời từ những máy chủ này.

#### Đặc điểm

Các đặc trưng cơ bản của HTTP:

* Kết nối không liên tục: Kết nối của HTTP không liên tục. Quy trình xử lý, phản hồi yêu cầu thông thường của HTTP là Client tạo yêu cầu -> dừng kết nối với Server để đợi phản hồi -> Server tiến hành xử lý yêu cầu -> thiết lập kết nối tới Client và gửi phản hồi.
* Độc lập: Đặc trưng cơ bản thứ hai của HTTP là tính độc lập. Bạn có thể gửi mọi loại dữ liệu qua HTTP miễn sao máy chủ và Client có biện pháp kiểm soát các nội dung của dữ liệu. Client và Server cần xác định nội dung gửi đi thuộc kiểu gì để lựa chọn MIME phù hợp.
* HTTP là stateless: Là connectionless nên đặc trưng thứ ba của HTTP là Stateless. Máy chủ và Client chỉ biết nhau trong yêu cầu của hiện tại, ngay sau đó, chúng sẽ quên. Ngoài ra, Cả máy khách và server đều có thể lưu lại thông tin về những yêu cầu giữa các website.

## Cơ sở dữ liệu

### So sánh cơ sở dữ liệu SQL và NoSQL

Có khá nhiều cơ sở có thể sử dụng như là SQL server, MongoDB, … trong đó cơ sở dữ liệu NoSQL cụ thể là MongoDB sẽ được sử dụng trong hệ thống bởi nó có ưu điểm như là dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ. Đặc biệt dữ liệu sẽ được caching (ghi đệm) lên RAM, hạn chế truy cập vào ổ cứng nên tốc độ đọc và ghi cao.

Sự so sánh về đặc điểm của hai cơ sở dữ liệu SQL và NoSQL được thể hiện rõ trong bảng 1-2 [3]. Từ đó, chúng ta có thể thấy được ưu điểm của cơ sở dữ liệu NoSQL.

Bảng . So sánh cơ sở dữ liệu SQL và NoSQL

|  |  |  |
| --- | --- | --- |
| **Tham số** | **SQL** | **NoSQL** |
| Định nghĩa | Cơ sở dữ liệu SQL chủ yếu được gọi là RDBMS hoặc Cơ sở dữ liệu quan hệ | Cơ sở dữ liệu NoSQL chủ yếu được gọi là cơ sở dữ liệu không liên quan hoặc phân tán |
| Ngôn ngữ Query | Structured query language (SQL) | Không có ngôn ngữ query |
| Loại hình | SQL databases là cơ sở dữ liệu dựa trên bảng | NoSQL databases có thể dựa trên tài liệu, cặp khóa-giá trị, cơ sở dữ liệu biểu đồ |
| Lược đồ | SQL databases có lược đồ được xác định trước | NoSQL databases sử dụng lược đồ động cho dữ liệu phi cấu trúc. |
| Khả năng mở rộng | SQL databases có thể mở rộng theo chiều dọc | NoSQL databases có thể mở rộng theo chiều ngang |
| Lưu trữ dữ liệu phân cấp | SQL databases không thích hợp cho việc lưu trữ dữ liệu phân cấp. | Phù hợp hơn cho kho lưu trữ dữ liệu phân cấp |
| Biến thể | Một loại có biến thể nhỏ | Nhiều loại khác nhau bao gồm các kho khóa-giá trị, cơ sở dữ liệu tài liệu và cơ sở dữ liệu đồ thị. |
| Loại lưu trữ | Highly Available Storage (SAN, RAID, etc.) | Commodity drives storage (standard HDDs, JBOD) |
| Tính năng nổi bật | Hỗ trợ đa nền tảng, Bảo mật và miễn phí | Dễ sử dụng, hiệu suất cao và công cụ linh hoạt. |

### Tổng quan về MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh [4].

Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng.

So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row, MongoDB sẽ dùng các document thay cho row trong RDBMS.

Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.

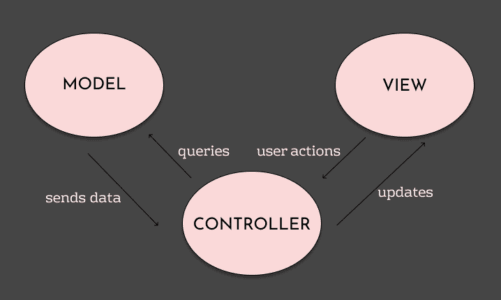
Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB.

## Tổng quan về mô hình Model-View-Controller (MVC)

### Khái niệm

MVC là viết tắt của cụm từ “Model-View-Controller”. MVC là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tính. MVC chia thành ba phần được kết nối với nhau được thể hiện trong hình 1-2. Theo hình 1-2, mô hình MVC bao gồm Model (dữ liệu), View (giao diện) và Controller (bộ điều khiển) [5]. Cụ thể về chức năng của từng phần như sau:

* Model: quản lý và xử lý cơ sở dữ liệu (database), logic và quy tắc (rule) cho hệ thống.
* View: View là phương tiện hiển thị các đối tượng trong một ứng dụng. Chẳng hạn như hiển thị một cửa sổ, nút hay văn bản trong một cửa sổ khác. Nó bao gồm bất cứ thứ gì mà người dùng có thể nhìn thấy được.
* Controller: là thành phần điều hướng, là chất kết dính giữa model và view, có nhiệm vụ nhận những request từ người dùng, tương tác với model để lấy thông tin và gửi cho view để hiển thị lại cho người dùng.



Hình . Mô hình MVC

Ở đây trong mô hình MVC, View sẽ không giao tiếp trực tiếp với Model mà sự tương tác giữa View và Model sẽ chỉ được xử lý bởi Controller.

### Đặc điểm của mô hình MVC

* Kiểm tra lỗi đơn giản và dễ dàng.
* Các phần Model, Controller và View được tách biệt với nhau.
* Tối ưu trên các nền tảng khác nhau.
* Dễ dàng bảo trì và phát triển ứng dụng vì các phần của ứng dụng được tách biệt với nhau.

## Giới thiệu về công cụ và kỹ thuật được sử dụng

### Ngôn ngữ C#

C# là một ngôn ngữ lập trình hướng đối tượng được phát triển bởi Tập đoàn Microsoft. Ngôn ngữ này được xem là ngôn ngữ lập trfinh hướng đối tượng trong sáng và thuần nhất. Nó hiện thực hầu hết các tính chất tốt của mô hình hướng đối tượng giống như ngôn ngữ lập trình Java. C# là ngôn ngữ được Microsoft phát triển dựa trên 2 ngôn ngữ huyền thoại đó là C++ và Java. Và nó cũng được miêu tả là loại ngôn ngữ có được sự cân bằng giữa C++, Visual Basic, Delphi và Java.

C# làm việc chủ yếu trên bộ khung .NET (.NET framework). Ngôn ngữ lập trình này có khả năng tạo ra nhiều ứng dụng mạnh mẽ và an toàn cho nền tảng Windows. Các thành phần máy chủ, dịch vụ web, ứng dụng di động và nhiều khả năng khác nữa.

C# là ngôn ngữ gần gũi với các ngôn ngữ lập trình thông dụng như C++, Java hay Pascal. Bởi nó được xây dưng trên nền tảng của những ngôn ngữ này. Chính bởi vậy mà nó sẽ được kế thừa những tinh hoa của những ngôn ngữ thông dụng trên. Khi C# ra đời nó đã giải quyết được các khiếm khuyết của C/C++ như con trỏ, các hiệu ứng phụ, … Vì vậy nó dễ tiếp cận và dễ được phát triển hơn.  Đây cũng là một lợi thế cho những ai đã học và quen thuộc với C++ hay Java khi mà học ngôn ngữ C#.

C # được hỗ trợ bởi một cộng đồng các nhà phát triển lớn. Chính cộng đồng sử dụng này đã khiến cho C# trở thành ngôn ngữ đáng tin cậy và hoạt động hiệu quả hơn.

C# có thể được sử dụng để tạo ra hầu hết mọi ứng dụng, Nhưng nó mạnh mẽ nhất là trong việc xây dựng ứng dụng và game trên desktop của Windows. C# cũng ngày càng trở nên phổ biến cho phát triển điện thoại di động và các ứng dụng web. Một số ví dụ về phát triển Game sử dụng bộ khung C# và [.NET](https://vi.wikipedia.org/wiki/.NET_Framework) là Unity, Mono Game và CryEngine.

### Visual Studio

#### Khái niệm Visual Studio

Visual Studio là một trong những công cụ hỗ trợ lập trình và thiết kế website rất nổi tiếng hiện nay của Microsoft và chưa có một phần mềm nào có thể thay thế được. Visual Studio được viết bằng 2 ngôn ngữ chính là C# và VB+. Đây là hai ngôn ngữ lập trình giúp người dùng có thể lập trình được hệ thống một các dễ dàng và nhanh chóng.

Visual Studio là một phần mềm lập trình hệ thống được sản xuất trực tiếp từ Microsoft. Từ khi ra đời đến nay, Visual Studio đã có rất nhiều các phiên bản sử dụng khác nhau. Điều đó, giúp cho người dùng có thể lựa chọn được phiên bản tương thích với dòng máy của mình cũng như cấu hình sử dụng phù hợp nhất [6].

#### Một số tính năng mềm Visual Studio

* Biên tập mã
* Giống như bất kỳ IDE khác, Visual Studio gồm có một trình soạn thạo mã hỗ trợ tô sáng cú pháp và hoàn thiện mã bằng cách sử dụng IntelliSense không chỉ cho các biến, hàm và các phương pháp mà còn các cấu trúc ngôn ngữ như vòng điều khiển hoặc truy vấn.
* Hỗ trợ cải tiến mã nguồn bao gồm tham số sắp xếp lại, biến và phương pháp đổi tên, khai thác và đóng gói giao diện các lớp thành viên bên trong những trạng thái giữa những thứ khác.
* Tính năng biên dịch nền (biên dịch gia tăng) Visual Studio biên dịch nó trong nền để cung cấp thông tin phản hồi về cú pháp và biên dịch lỗi, được đánh dấu bằng một gạch dưới gợn sóng màu đỏ.
* Trình gỡ lỗi
* Trình gỡ lỗi hoạt động vừa là một trình gỡ lỗi cấp mã nguồn và là một trình gỡ lỗi cấp máy. Hoạt động với cả hai mã quản lý cũng như ngôn ngữ được hỗ trợ bởi Visual Studio. Ngoài ra, cũng có thể đính kèm theo quy trình hoạt động và theo dõi và theo dõi những quy trình. Các chương trình đa luồng cao cấp cũng được hỗ trợ.
* Trình gỡ lỗi cho phép thiết lập các breakpoint và watch (giúp giám sát các giá trị của biến). Breakpoint có thể có điều kiện, nghĩa là chúng được kích hoạt khi điều kiện được đáp ứng. Khi gỡ lỗi, nếu con trỏ chuột di chuyển lên bất kì biến, giá trị hiện tại của nó được hiển thị trong phần chú giải. Trong quá trình viết mã, các trình gỡ lỗi của visual Studio cho phép một số chứng năng được gọi ra bằng tay từ cửa sổ cung cụ Immediate.

#### Tính năng thiết kế

* WindowsForm Designer: được sử dụng với mục đích xây dựng GUI sử dụng WindowsForms, được bố trí dùng để xây dựng các nút điều khiển bên trong hoặc cũng có thể khóa chúng vào bên cạnh mẫu. Điều khiển trình bày dữ liệu có thể được liên kết với các nguồn dữ liệu như: cơ sở dữ liệu hoặc truy vấn.
* WPF Desidner: tính năng này giống như WindowsForms Designer có công dụng hỗ trợ kéo và thả. Sử dụng tương tác giữa người và máy tính nhằm múc tiêu theo Windows Presentation Foundation. Hỗ trợ các chức năng WPF bao gồm kết nối dữ liệu và tự động hóa bố trí quản lý. Tao ra mã XAML cho giao diện người dùng. Các tập tin XAML được tạo ra là tương thích với Microsoft Expression Degin, sản phẩm thiết kế theo định hướng. Các mã XAML được liên kết với mã đang sử dụng một mô hình code – behind.
* Web designer/ development: Visual Studio cũng có một trình soạn thảo và thiết kế website cho phpes các trang web được thiết kế theo tính năng kéo và thả đối tượng
* Xamarin: công cụ được thêm vào để xây dựng các ứng dụng di dộng đa nền tảng.

#### Các công cụ khác

* Open Tabs Browser: được sử dụng để liệt kê tất cả thẻ đang mở và chuyển đổi giữa chúng.
* Properties Editor: được sử dụng để chỉnh sửa các thuộc tính trong một cửa sổ giao diện bên trong Visual Studio.
* Object Browser: là một không gian tên và trình duyệt lớp thư viện cho Microsoft NET. Có thể được sử dụng để duyệt các không gian tên (được sắp xếp theo thứ bậc) trong Assembly (CLI).
* Solution Explorer: theo cách nói trong Visual Studio, là một tập hợp các tập tin mã và các nguồn khác được sử dụng để xây dựng một ứng dụng
* Team Explorer: được sử dụng để tích hợp các khả năng của Team Foundation Server, Revision Control System và là cơ sở cho môi trường CodePlex đối với dự án mã nguồn mở.
* Visual Studio Tools for Office: là một SDK và một add – in cho Visual Studio bao gồm các công cụ để phát triển cho các bộ Microsoft Office

### Visual Studio Code

Visual Studio Code là một trong những trình soạn thảo mã nguồn rất phổ biến được các lập trình viên sử dụng. Với các ưu điểm nổi bật là sự nhanh chóng, nhẹ, hỗ trợ đa nền tảng cùng nhiều tính năng và là mã nguồn mở chính.

### ReactJs

ReactJS là một opensource được phát triển bởi Facebook, ra mắt vào năm 2013, bản thân nó là một thư viện Javascript được dùng để để xây dựng các tương tác với các thành phần trên website. Một trong những điểm nổi bật nhất của ReactJS đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client nữa.

### Application Programming Interface (API)

API là các phương thức, giao thức kết nối với các thư viện và ứng dụng khác. Nó là viết tắt của Application Programming Interface – giao diện lập trình ứng dụng. API cung cấp khả năng truy xuất đến một tập các hàm hay dùng. Và từ đó có thể trao đổi dữ liệu giữa các ứng dụng. API hiện nay đều tuân thủ theo tiêu chuẩn REST và HTTP, tạo sự thân thiện dễ sử dụng với nhà phát triển. Giúp người dùng dễ dàng truy cập, dễ hiểu hơn.

Web API là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.

## Kết luận

Như vậy, chương một đã trình bày về lý do lựa chọn đề tài, mô hình tổng quan của hệ thống cảnh báo cháy, thể hiện qua sơ đồ và mô tả rõ hoạt động từng thành phần của hệ thống, sau đó giới thiệu một số giải pháp và công cụ được sử dụng trong đề tài này. Kết thúc chương 1 “Tổng quan hệ thống cảnh báo cháy”, chúng ta đã có thể hiểu được sơ bộ về đề tài, và tiếp theo đây chương 2 sẽ đi vào phân tích thiết kế chi tiết cho hệ thống.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

Sau khi đã xác định được mục đích xây dựng ứng dụng đã nêu ở chương 1, chúng ta tiếp tục phân tích các chức năng cần thiết của ứng dụng. Từ đó tạo ra sơ đồ Use Case, các quan hệ trong lược đồ và mô tả lại từng yêu cầu hệ thống theo một kịch bản. Sau đó tìm hiểu thiết kế biểu đồ lớp (Class Diagram) cho Models chung và cuối cùng là thiết kế luồng đi của hệ thống, luồng đi của dữ liệu thông qua biểu đồ hoạt động (Activity Diagram) và biểu đồ tuần tự (Sequence Diagram).

## Yêu cầu hệ thống

### Yêu cầu chức năng

Chức năng cơ bản của hệ thống:

* Xem danh sách các căn hộ, danh sách thiết bị.
* Xem các chỉ số trong ngày của các căn hộ đo được từ cảm biến, xem các chỉ số được tổng hợp lại theo giờ trong một ngày.

### Yêu cầu phi chức năng

* Các ứng dụng chạy ổn định
* Giao diện dễ nhìn, phông chữ thống nhất
* Dễ sử dụng
* Ít lỗi
* Dễ sửa chữa khi có lỗi

## Xây dựng Use-case Diagram

### Tìm hiểu biểu đồ Use case

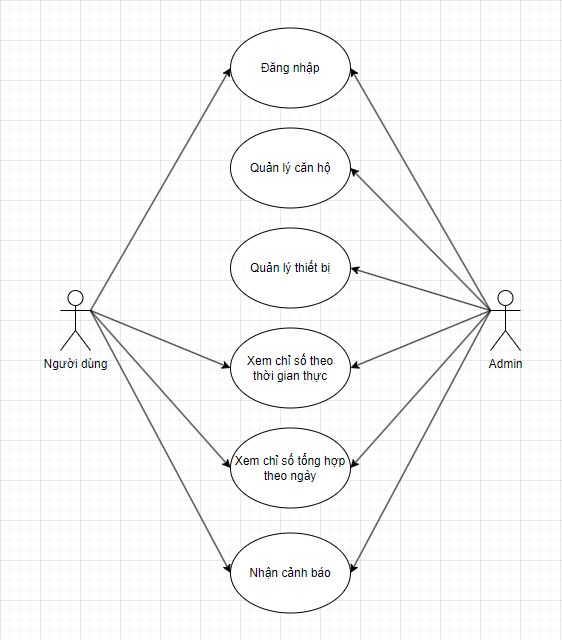
Biểu đồ Use Case biểu diễn sơ đồ chức năng của hệ thống. Từ tập yêu cầu của hệ thống, biểu đồ use case chỉ ra những điều hệ thống cần thực hiện để thỏa mãn các yêu cầu của người dùng hệ thống đó 🡪 Chỉ ra sự tương tác giữa tác nhân và hệ thống thông qua các Use Case. Các thành phần cơ bản của một biểu đồ Use case được thể hiện trong bảng 2-1.

Bảng . Các phần tử của biểu đồ Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **Phần tử mô hình** | **Ý nghĩa** | **Cách biểu diễn** | **Ký hiệu trong biểu đồ** |
| Use case | Biểu diễn một chức năng xác định của hệ thống | Hình Elip chứa tên của Use Case |  |
| Tác nhân | Là một đối tượng bên ngoài hệ thống tương tác trực tiếp với các use case. | Biểu diễn bởi một lớp kiểu Actor. | A picture containing chart  Description automatically generated |
| Mối quan hệ giữa các use case | Include: khi thực thi Use Case A sẽ luôn cần thực hiện Use Case B. | Extend và include có dạng các mũi tên đứt nét. | Text  Description automatically generated with low confidence  Text  Description automatically generated with low confidence |
| Extend: khi thực thi Use Case A, trong một số trường hợp sẽ thực hiện Use Case B. |
| Comunication Link: Kết nối giữa actor và use case, cho biết actor đó có những tương tác nào đến hệ thống | Comunication Link có dạng đường thẳng |
| Generalization: user case này được thiết kế các chức năng từ use case kia. | Generalization có dạng mũi tên tam giác. |
| Biên của hệ thống | Tách biệt phần bên trong và bên ngoài hệ thống. | Được biểu diễn bởi một hình chữ nhật rỗng. |  |

### Bản vẽ Use case

#### Use case tổng quát



Hình . Sơ đồ Use Case tổng quát của hệ thống

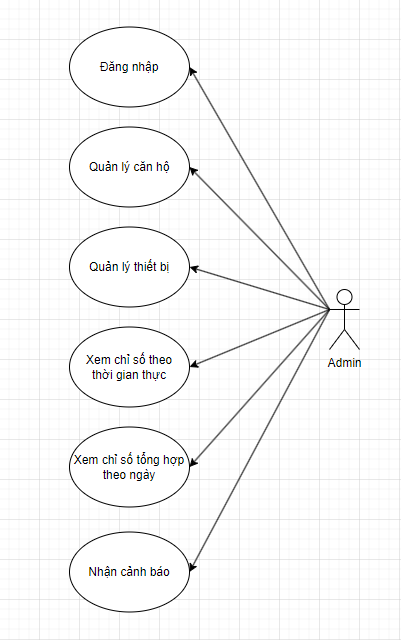
Mô tả:

Hệ thống gồm hai tác nhân chính: admin và người dùng.

Bác sĩ là người có quyền sử dụng toàn bộ chức năng của hệ thống.

Người dùng là chủ các căn hộ, có thể xem các chỉ số của căn hộ và nhận cảnh báo nếu có bất thường với các chỉ số đó.

* Use case cho Actor “Admin”

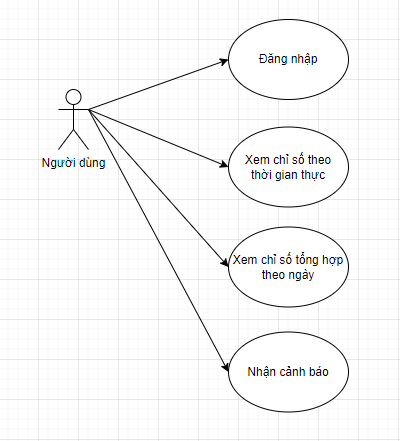


Hình . Sơ đồ use case cho admin

Mô tả sơ đồ:

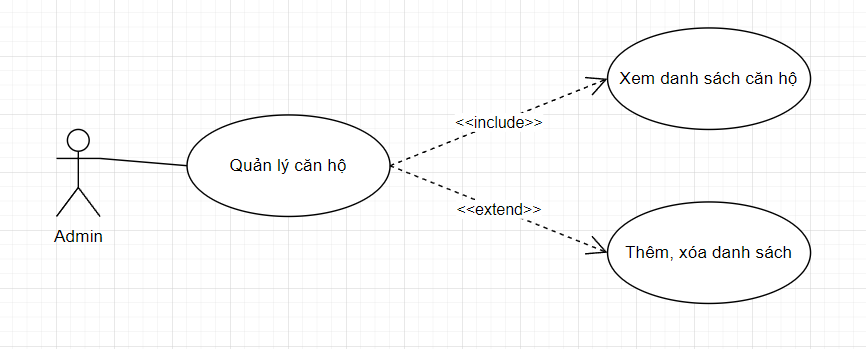
Admin là tác nhân có thể sử dụng được toàn bộ chức năng của hệ thống, quản lý các căn hộ, thiết bị và có thể xem và nhận cảnh báo về các bất thường trong chỉ số của các căn hộ.

* Use case cho Actor “Người dùng”



Hình . Sơ đồ use case cho người dùng

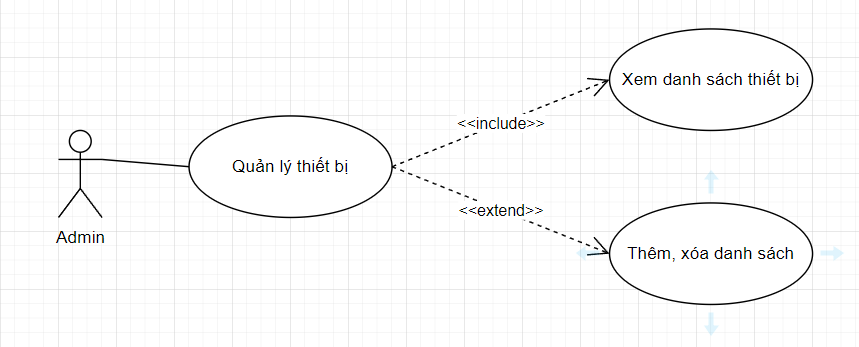
#### Usecase quản lý bệnh nhân



Hình . Usecase quản lý căn hộ

Mô tả: Chức năng quản lý căn hộ được sử dụng bởi actor admin có thể quản lý toàn bộ danh sách căn hộ.

#### Usecase quản lý thiết bị



Hình . Usecase quản lý thiết bị

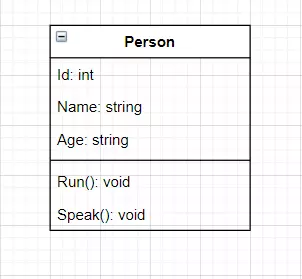
Mô tả: Chức năng quản lý thiết bị bao gồm xem danh sách thiết bị và có thể chỉnh sửa danh sách đó.

## Xây dựng Class Diagram

### Tìm hiểu biểu đồ lớp (Class Diagram)

Class diagram là một kỹ thuật mô hình hóa tồn tại ở tất cả các phương pháp phát triển hướng đối tượng mô tả kiểu của các đối tượng trong hệ thống và các loại quan hệ khác nhau tồn tại giữa chúng. Class diagram là biểu đồ hay dùng nhất trong UML và gần gũi nhất với các lập trình viên, giúp các lập trình viên trao đổi với nhau và hiểu rõ ý tưởng của nhau [8].

Các tính chất cơ bản của class diagram Ví dụ trong hình 2.7:



Hình . Ví dụ về các tính chất cơ bản của class

* Tên class: Person.
* Attribute (field, property): Id, Name, Age.
* Operation (method, function): Run(), Speak().

Access Modifier trong class diagram: Sử dụng để đặc tả phạm vi truy cập cho các Attribute và Operation của 1 class. Có 4 lựa chọn phạm vi truy cập được thể hiện trong hình 2.8, chi tiết như sau:

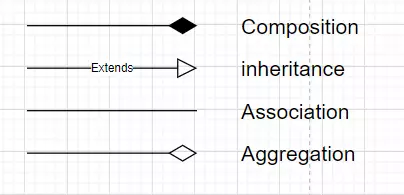
Table

Description automatically generated

Hình . Ví dụ về các phạm vi truy cập của class

* Private (-): Chỉ mình các đối tượng được tạo từ class này có thể sử dụng.
* Public (+): Mọi đối tượng đều có thể sử dụng.
* Protected (#): Chỉ các đối tượng được tạo từ class này và class kế thừa từ class này có thể sử dụng.
* Package/Default: Các đối tượng được tạo từ class trong lớp cùng gói có thể sử dụng.

Relationship trong class diagram: Sử dụng để thể hiện mỗi quan hệ giữa đối tượng được tạo từ 1 class với các đối tượng được tạo từ class khác trong class diagram. Có 4 loại relationship được thể hiện trong hình 2.9. Trong đó:



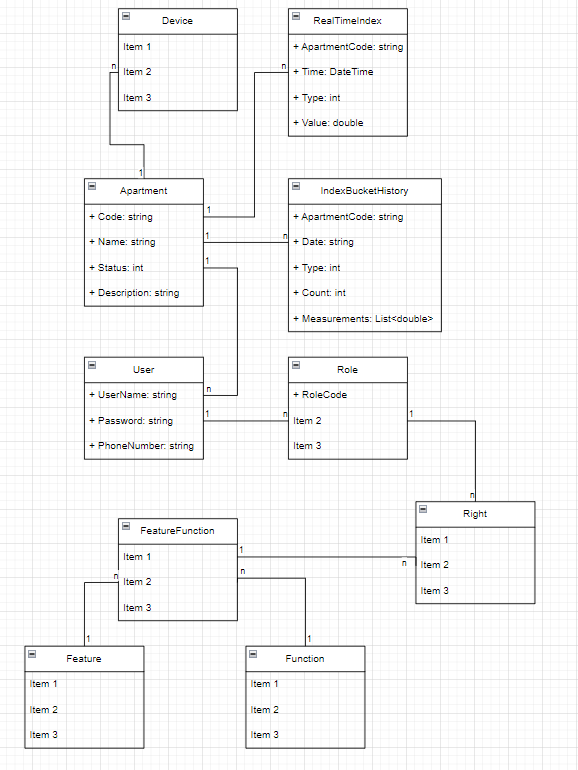
Hình . Các loại quan hệ trong class diagram

* Inheritance: 1 class kế thừa từ 1 class khác.
* Association: 2 class có liên hệ với nhau nhưng không chỉ rõ mối liên hệ.
* Composition: Đối tượng tạo từ class A mất thì đối tượng tạo từ class B sẽ mất.
* Agreegation: Đối tượng tạo từ class A mất thì đối tượng tạo từ class B vẫn tồn tại độc lập.

Multiplicity trong class diagram: Sử dụng để thể hiện quan hệ về số lượng giữa các đối tượng được tạo từ các class trong class diagram. Các kiểu multiplicity:

* 0...1: 0 hoặc 1.
* n: Bắt buộc có n.
* 0...\*: 0 hoặc nhiều.
* 1...\*: 1 hoặc nhiều.
* m... n: có tối thiểu là m và tối đa là n.

### Bản vẽ Class Diagram



Hình . Bản vẽ class diagram cho Model

## Xây dựng Activity Diagram

### Tìm hiểu biểu đồ hoạt động (Activity Diagram)

Activity diagram (biểu đồ hoạt động) là một mô hình logic được dùng để mô hình hoá cho các hoạt động trong một quy trình nghiệp vụ. Nó chỉ ra luồng đi từ hoạt động này sang hoạt động khác trong một hệ thống. Nó đặc biệt quan trọng trong việc xây dựng mô hình chức năng của hệ thống và nhấn mạnh tới việc chuyển đổi quyền kiểm soát giữa các đối tượng [9].

Để xây dựng được 1 bản vẽ activity diagram thì cần có những bước cơ bản như sau:

Bước 1: Xác định nghiệp vụ cần mô tả

Dựa vào các bản vẽ use case để xác định nghiệp vụ nào cần mô tả.

Bước 2: Xác định trạng thái đầu tiên và trạng thái kết thúc

Bước 3: Xác định các hoạt động tiếp theo

Xuất phát từ điểm bắt đầu, phân tích để xác định các hoạt động tiếp theo cho đến khi gặp điểm kết thúc để hoàn tất bản vẽ này.

Các thành phần của activity diagram

1. Start

* Ký hiệu:
* Đặc trưng:
  + - Khởi tạo một hoạt động
    - Một activity diagram có thể có nhiều trạng thái start
* Nên đặt tên là động từ. Và mô tả đủ ý nghĩa tổng thể của hoạt động có thể

1. Transition

* Ký hiệu:
* Đặc trưng: Mô tả sự chuyển đổi trạng thái của các hoạt động.

1. Decisition

* Ký hiệu:
* Đặc trưng:
  + Tập các điều kiện kích hoạt việc chuyển trạng thái
  + Branch
    - Mô tả điều kiện rẽ nhánh
    - Chỉ một dòng điều khiển đi vào
    - Hai hoặc nhiều dòng điều khiển ra
    - Chỉ một dòng điều khiển ra dẫn đến kết quả
    - Mỗi dòng chứa một điều kiện (guard), guard phải liên quan đến điều kiện và loại trừ nhau
  + Merge
    - Có hai hoặc nhiều dòng điều khiển đi vào
    - Chỉ một dòng điều khiển đi ra
* Có thể hiểu đây là ký hiệu biểu thị nút điều kiện chuyển hướng. Tùy theo trường hợp đúng sai của kết quả biểu thức logic bên trong ký hiệu mà có hướng di chuyển tiếp theo tương ứng.

1. Synchronization bar

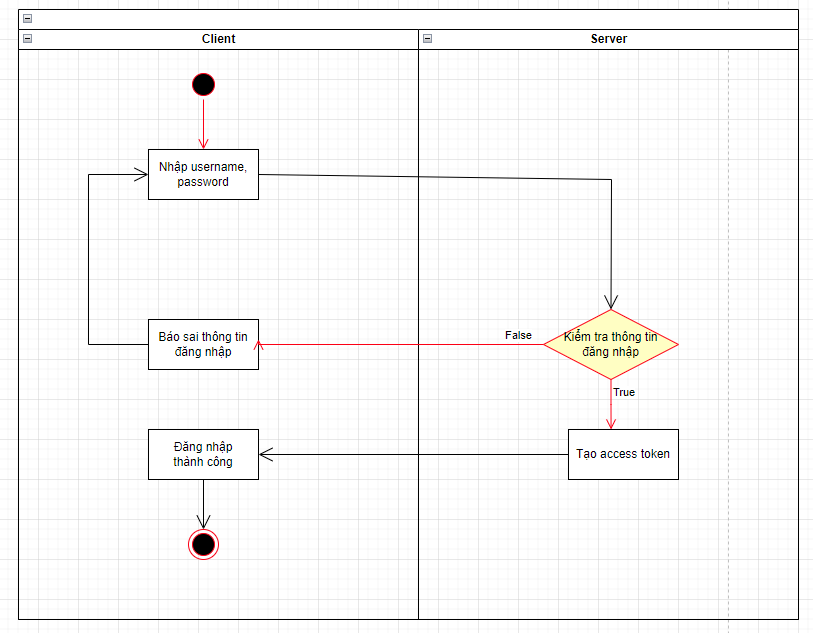
* Ký hiệu:
* Đặc trưng:
  + Mô tả dòng điều khiển thực hiện song song
  + Fork
    - Mô tả một dòng điều khiển được tách ra thực hiện song song
    - Chỉ một dòng điều khiển đi vào
    - Có hai hoặc nhiều dòng điều khiển đi ra
    - Dùng fork khi các hoạt động thực hiện không quan tâm thứ tự
  + Join
    - Kết hợp các dòng điều khiển song song
    - Có hai hoặc nhiều dòng điều khiển vào
    - Chỉ một dòng điều khiển ra
    - Dòng điều khiển ra được tạo khi tất cả các dòng cần thiết đã vào
* Chú ý: Fork và Join không cần nhãn

1. End

* Ký hiệu:
* Đặc trưng:
  + Mô tả trạng thái kết thúc quy trình
  + Một activity diagram có một hoặc nhiều trạng thái kết thúc
* Là điểm kết thúc luồng.

### Bản vẽ Activity Diagram

#### Biểu đồ hoạt động chức năng đăng nhập



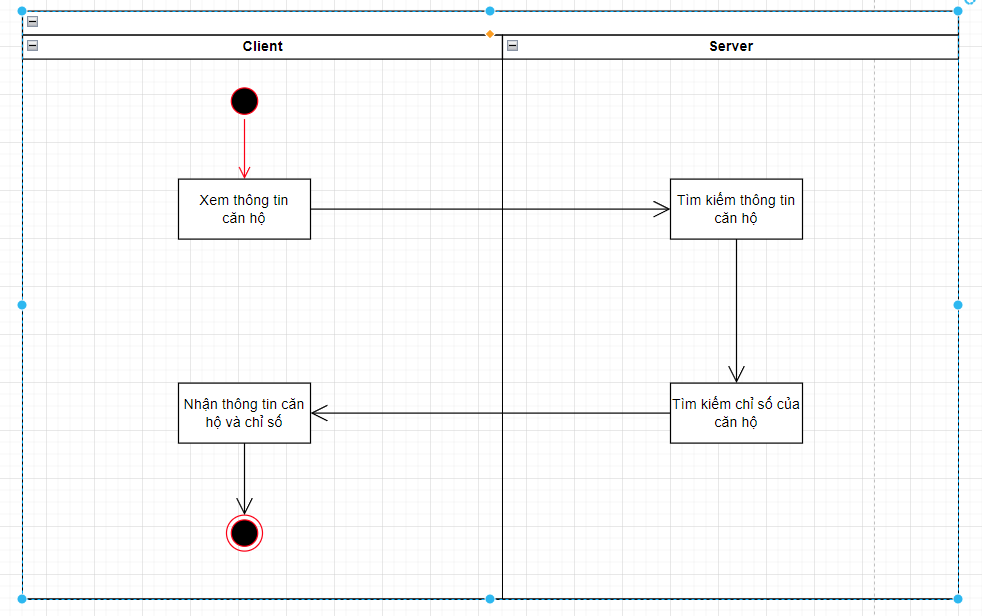
Hình . Biểu đồ hoạt động chức năng đăng nhập

Mô tả biểu đồ:

Để sử dụng hệ thống người dùng cần đăng nhập tài khoản mà đã được admin tạo sẵn. Hệ thống phải được kết nối internet. Khi có kết nối người dùng sẽ vào phần đăng nhập sau đó nhập 2 thành phần UserName và Password.

Khi Nhập UserName và Password, phía hệ thống sẽ kiếm tra thông tin đăng nhập, nếu không tồn tại sẽ có một thông báo “Sai tên đăng nhập hoặc mật khẩu” và yêu cầu nhập lại, khi thông tin đăng nhập đúng thì người dùng đăng nhập thành công.

#### Biểu đồ hoạt động chức năng xem chỉ số căn hộ



Hình . Biểu đồ hoạt động chức năng xem chỉ số căn hộ

Mô tả biểu đồ:

Để sử dụng chức năng xem chỉ số, người dùng phải được kết nối với hệ thống thông qua chức năng đăng nhập. Hệ thống phải được duy trì kết nối internet. Khi đó người dùng sẽ vào phần theo dõi chỉ số trong Client để tiến hành xem chỉ số.

Khi vào phần theo dõi chỉ số, phía hệ thống sẽ truy cập vào cơ sở dữ liệu nơi mà các chỉ số được cập nhật liên tục bởi các thiết bị đo. Sau đó, hệ thống sẽ lấy các dữ liệu về chỉ số ra và gửi lại cho Client của người dùng.

#### Biểu đồ hoạt động chức năng cảnh báo

Diagram

Description automatically generated

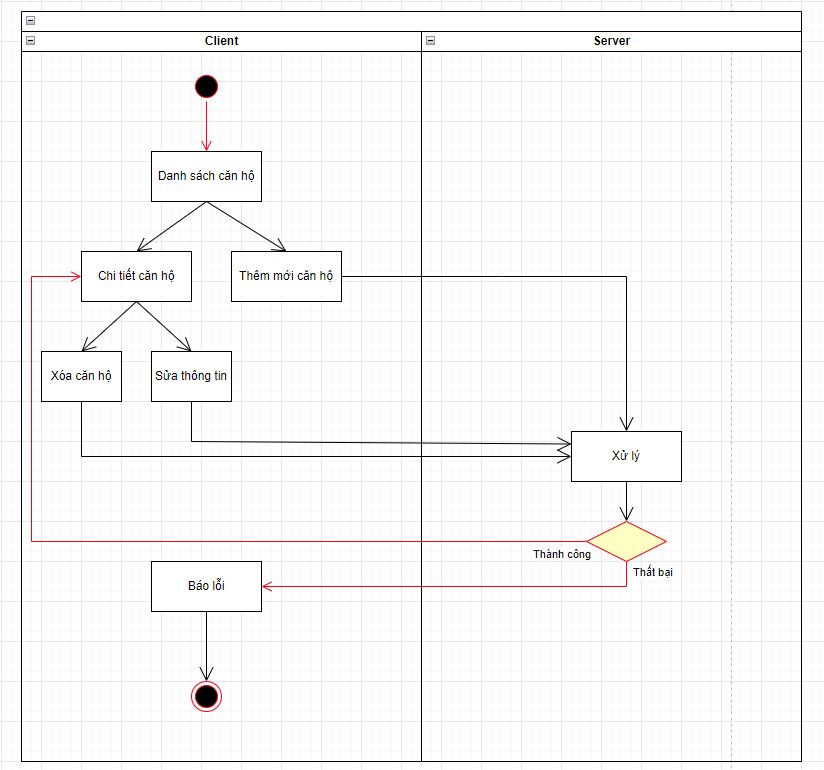
Hình . Biểu đồ hoạt động chức năng cảnh báo

Mô tả biểu đồ:

Để sử dụng chức năng nhận cảnh báo, người dùng phải được kết nối với hệ thống thông qua chức năng đăng nhập. Hệ thống phải được duy trì kết nối internet. Khi đó người dùng sẽ nhận được cảnh báo nếu chỉ số có bất thường.

Thiết bị đo sẽ liên tục cập nhật dữ liệu về các chỉ số vào Database, bản thân thiết bị cũng có thể phát hiện ra các chỉ số bất thường và phát ra cảnh báo, đồng thời gửi một thông báo về bất thường đến cho Server, Server sẽ xử lý và gửi cảnh báo đến cho những User tương ứng.

#### Biểu đồ hoạt động chức năng quản lý căn hộ



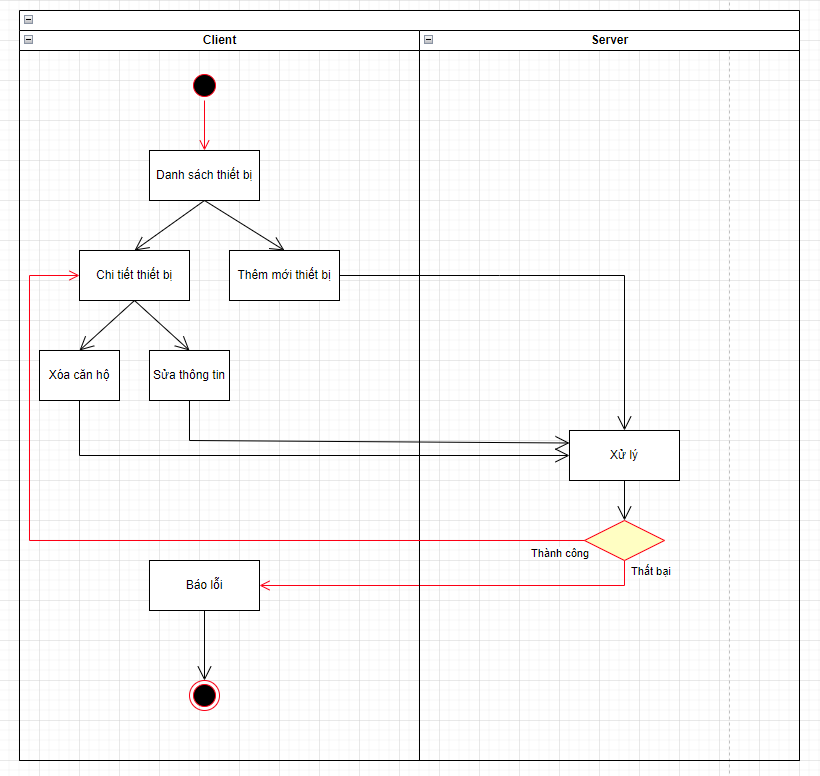
Hình . Biểu đồ hoạt động chức năng quản lý căn hộ

Mô tả biểu đồ:

Để sử dụng chức năng quản lý căn hộ, người dùng phải được kết nối với hệ thống thông qua chức năng đăng nhập. Hệ thống phải được duy trì kết nối internet.

Người dùng có thể xem danh sách các căn hộ, sau đó thực hiện các hành động xem chi tiết, thêm, xóa, sửa căn hộ. Server sau khi xử lý, nếu thành công sẽ quay lại màn hình chi tiết căn hộ, nếu thất bại sẽ hiển thị thông báo “Thất bại” và kết thúc luồng.

#### Biểu đồ hoạt động chức năng quản lý thiết bị



Hình . Biểu đồ hoạt động chức năng quản lý thiết bị

Mô tả biểu đồ:

Để sử dụng chức năng quản lý thiết bị, người dùng phải được kết nối với hệ thống thông qua chức năng đăng nhập. Hệ thống phải được duy trì kết nối internet.

Người dùng có thể xem danh sách các thiết bị, sau đó thực hiện các hành động xem chi tiết, thêm, xóa, sửa thiết bị. Server sau khi xử lý, nếu thành công sẽ quay lại màn hình chi tiết thiết bị, nếu thất bại sẽ hiển thị thông báo “Thất bại” và kết thúc luồng.

## Xây dựng Sequence Diagram

### Tìm hiểu biểu đồ tuần tự (Sequence Diagram)

Biểu đồ tuần tự là biểu đồ dùng để xác định các trình tự diễn ra sự kiện của một nhóm đối tượng nào đó.

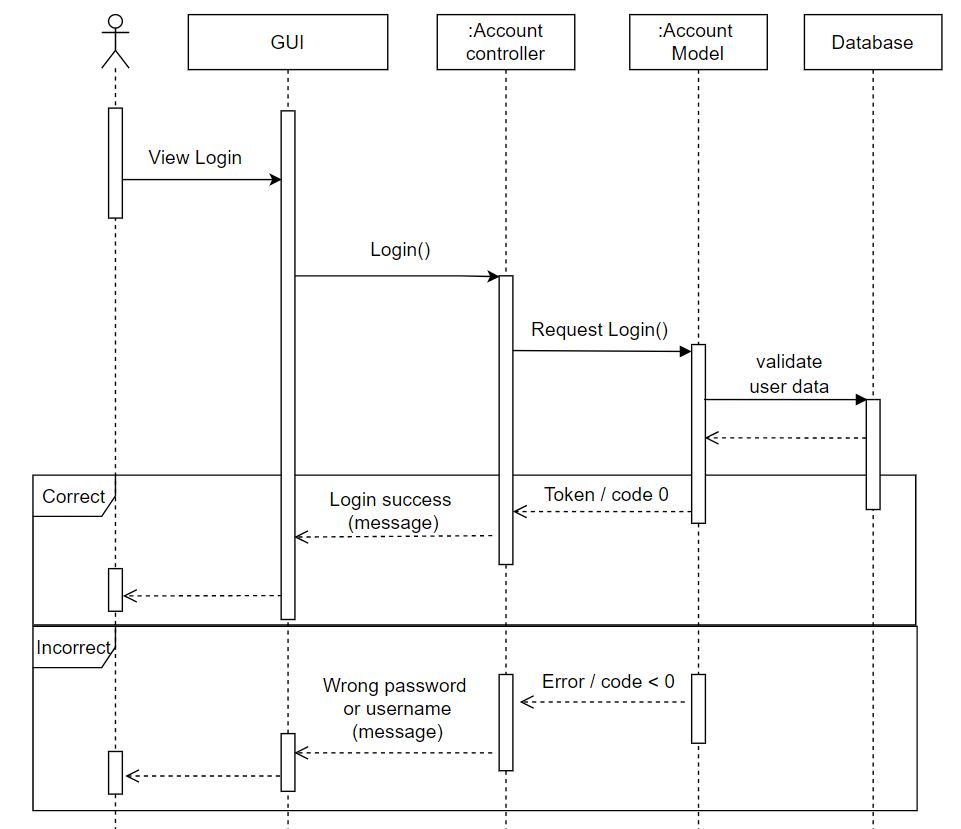
Nó mô tả chi tiết các thông điệp được gửi và nhận giữa các đối tượng đồng thời cũng chú trọng đến việc trình tự về mặt thời gian gửi và nhận các thông điệp đó. Các thành phần cơ bản của một biểu đồ tuần tự được thể hiện trong bảng 2-2.

Bảng . Các thành phần của biểu đồ tuần tự

|  |  |  |
| --- | --- | --- |
| **Tên thành phần** | **Ý nghĩa** | **Ký hiệu** |
| Đối tượng |  |  |
| Đường đời đối tượng | Biểu diễn bằng các đường gạch đứt thẳng đứng bên dưới đối tượng |  |
| Thông điệp đồng bộ | Thông điệp cần có một Request trước hành động tiếp theo | Text  Description automatically generated with low confidence |
| Thông điệp không đồng bộ | Thông điệp không cần có một Request trước hành động tiếp theo |  |
| Thông điệp chính mình | Thông điệp mà đối tượng gửi cho chính nó để thực hiện các hàm nội tại | Text  Description automatically generated with medium confidence |
| Thông điệp trả lời hoặc trả về | Thông điệp trả lời lại khi có Request hoặc sau khi kiểm tra tính đúng đắn của một điều kiện nào đó | Shape  Description automatically generated with medium confidence |

### Bản vẽ Sequence Diagram

#### Biểu đồ tuần tự chức năng đăng nhập

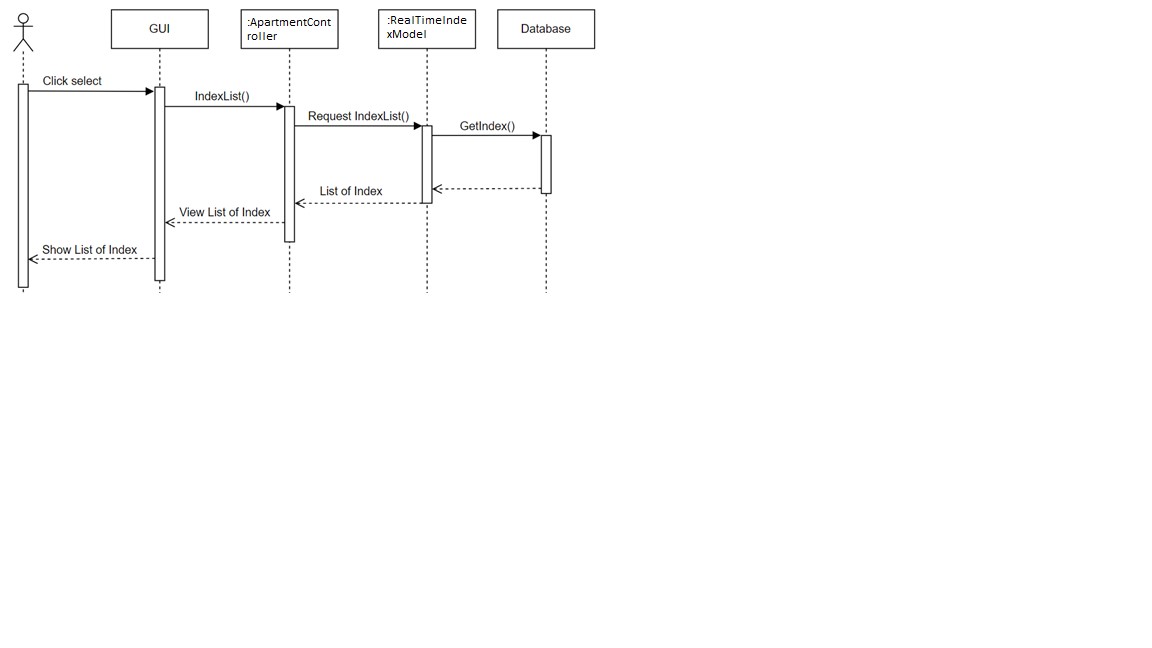


Hình . Biểu đồ tuần tự chức năng đăng nhập

Mô tả biểu đồ:

Khi người dùng tương tác với giao diện đăng nhập, nhập thông tin đăng nhập rồi ấn button đăng nhập, giao diện sẽ gọi đến hàm Login() trong AccountController. Controller này sẽ tạo ra một request về phía server để yêu cầu đăng nhập, server sẽ điều phối yêu cầu đó vào hàm Login() trong Account Model, nếu thông tin đăng nhập đúng, server sẽ trả về token và giá trị code 0 trong gói tin response, ngược lại nếu thông tin đăng nhập sai, server sẽ trả về các giá trị code < 0 cho client, từ đây client sẽ đưa ra thông báo cho người dùng nhận biết.

#### Biểu đồ tuần tự chức năng xem chỉ số căn hộ



Hình . Biểu đồ tuần tự chức năng xem chỉ số căn hộ

Mô tả biểu đồ: User chọn căn hộ, View gọi đến hàm IndexList() của ApartmentController, sau đó Controller này sẽ gửi request IndexList đến Sever. Server điều phối đến hàm IndexList() trong RealTimeIndex Model, sau đó trả về cho client.

#### Biểu đồ tuần tự chức năng cảnh báo

Diagram

Description automatically generated

Hình . Biểu đồ tuần tự chức năng cảnh báo

Mô tả biểu đồ: Thiết bị liên tục yêu cầu cập nhật chỉ số đến Server, nếu thiết bị phát hiện bất thường sẽ gửi một request Warning() đến cho Server và chính nó cũng phát ra cảnh báo. Server khi nhận được request Warning() sẽ tiến hành tìm các người dùng cần được cảnh báo và gửi đến WarningController phía client, Controller này sẽ đưa ra giao diện thông báo bất thường đến những người dùng cần thiết.

## Kết luận

Như vậy, chương hai đã trình bày về việc phân tích yêu cầu của hệ thống, đưa ra những bản vẽ thiết kế cho toàn hệ thống nói chung và từng chức năng nói riêng, bao gồm các thiết kế UML như: sơ đồ Use case, Class Diagram, Activity Diagram, Sequence Diagram. Cuối cùng, chương 2 đưa ra thiết kế chung cho giao thức liên lạc giữa các bên Client và Server. Kết thúc chương 2 “Phân tích thiết kế hệ thống”, chúng ta đã có thể hiểu được chi tiết hơn về yêu cầu đề tài, làm rõ những chức năng cơ bản của hệ thống, và tiếp theo đây chương 3 sẽ đi vào triển khai hệ thống, tiến hành thực hiện một số chức năng cơ bản của hệ thống cảnh báo cháy.

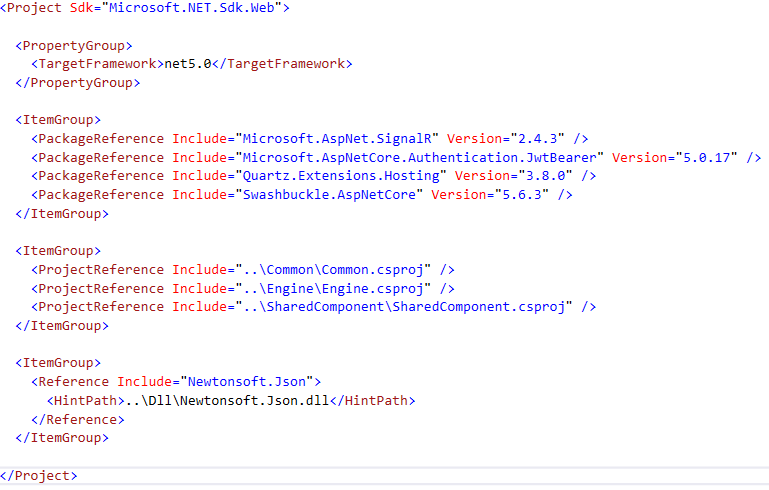
# TRIỂN KHAI HỆ THỐNG

Sau khi đã phân tích yêu cầu của hệ thống, đưa ra những bản vẽ thiết kế cho toàn hệ thống nói chung và từng chức năng nói riêng ở chương 2, chúng ta sẽ tiến hành triển khai ứng dụng cảnh báo cháy bằng mô hình MVC với công cụ Xamarin, Visual Studio. Mở đầu chương chúng ta sẽ tìm hiểu rõ hơn về Xamarin và Xamarin Forms, rồi sau đó tiến hành tạo project và triển khai một số mã nguồn cơ sở hỗ trợ cho việc tạo giao diện, cuối cùng là kết quả của một số chức năng đã thực hiện được.

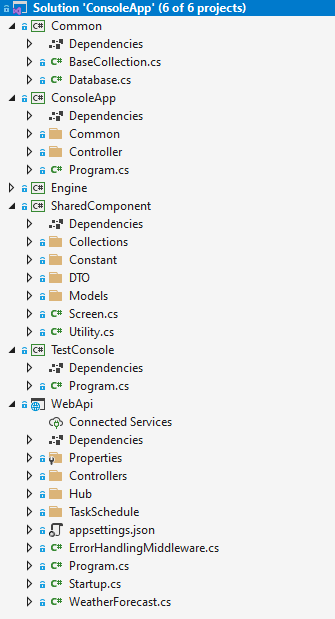
## Triển khai hệ thống

### Triển khai server hệ thống

Server của hệ thống được viết bằng ngôn ngữ C#, framework .Net Core 5.0



Hình . Framework và reference của Web API Server



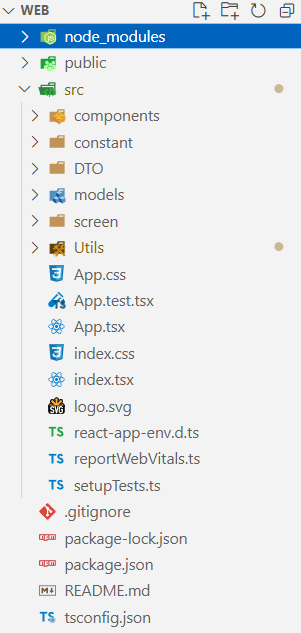
Hình . Cấu trúc solution API Web Server

Mô tả tổng quan các project trong solution Server hệ thống:

* Common: chứa các thành phần chung của các project, như là BaseCollection và Database để kết nối đến database MongoDB.
* ConsoleApp: là MQTT Server để nhận thông tin chỉ số của các căn hộ và lưu xuống database.
* SharedComponent: chứa các khai báo hằng số, enum (Constant); các lớp dùng để truyền dữ liệu (DTO); các lớp thực thể (Models); các hàm tiện ích xử lý chung (Utility).
* TestConsole: do đồ án chỉ triển khai một thiết bị đại diện để gửi thông tin của một căn hộ, nên project này dùng để giả lập các thiết bị sẽ liên tục gửi chỉ số của các căn hộ khác về MQTT Server.
* WebApi: Web Server cung cấp các đầu api để web client gửi yêu cầu, xử lý các yêu cầu và phản hổi cho web client.

### Triển khai web client

Web Client của hệ thống được xây dựng bằng framework ReactJS.



Hình . Cấu trúc của web client

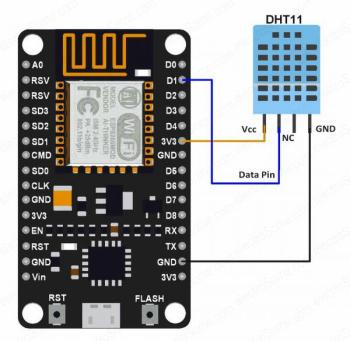
Mô tả tổng quan các thành phần trong project web client của hệ thống:

* Components: chứa các component như MyTextBox, MyComboBox, MyMessageBox, LoadingModal, …
* Constant: chứa các khai báo hằng số và enum; url của các api mà web server cung cấp.
* DTO: các lớp dùng để truyền dữ liệu (DTO) tương ứng với web server.
* Models: các lớp thực thể.
* Screen: mã nguồn các màn hình.
* Utils: các lớp tiện ích như các phương thức gửi yêu cầu đến server qua giao thức HTTP (HttpUtils), các phương thức để tương tác với local storage (useLocalStorage)

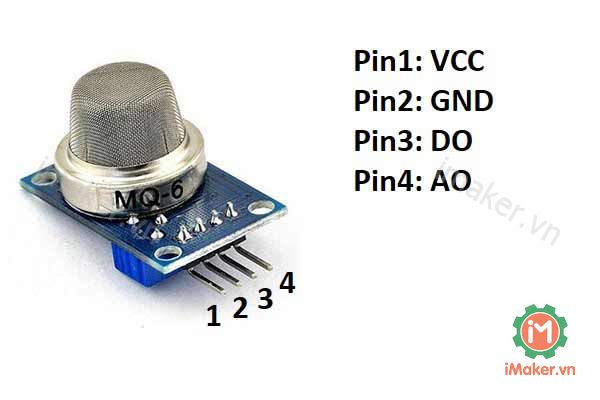
### Triển khai code phần cứng các thiết bị

Mã nguồn cho các thiết bị được viết bằng ngôn ngữ C++, compile và nạp code với Arduino IDE. Dưới đây là code để đọc chỉ số từ các cảm biến DHT11 và MQ6, sau đó gửi các chỉ số qua giao thức MQTT với ESP8266.

|  |
| --- |
| #include <ESP8266WiFi.h>  #include <PubSubClient.h>  #include "DHT.h"  #define DHTTYPE DHT11 // DHT 11  static char data[] = "";  const int DHTPin = 4;  DHT dht(DHTPin, DHTTYPE);     // Initialize DHT sensor.  static char celsiusTemp[7];  static char humidityTemp[7];  static char gasTemp[7];  static char agasTemp[7];  // Update these with values suitable for your network.  const char\* ssid = "Huy";  const char\* password = "h150201H";  const char\* mqtt\_server = "broker.emqx.io";  WiFiClient espClient;  PubSubClient client(espClient);  unsigned long lastMsg = 0;  #define MSG\_BUFFER\_SIZE (50)  char msg[MSG\_BUFFER\_SIZE];  int value = 0;  void setup\_wifi() {    delay(10);    // We start by connecting to a WiFi network    Serial.println();    Serial.print("Connecting to ");    Serial.println(ssid);    WiFi.mode(WIFI\_STA);    WiFi.begin(ssid, password);    while (WiFi.status() != WL\_CONNECTED) {      delay(500);      Serial.print(".");    }    randomSeed(micros());    Serial.println("");    Serial.println("WiFi connected");    Serial.println("IP address: ");    Serial.println(WiFi.localIP());  }  void warning() {      tone(12,220,125);      delay(125);      tone(12,2093,250);      delay(250);      tone(12,82,125);      delay(125);  }  void callback(char\* topic, byte\* payload, unsigned int length) {    warning();  }  void reconnect() {    // Loop until we're reconnected    while (!client.connected()) {      Serial.print("Attempting MQTT connection...");      // Create a random client ID      String clientId = "ESP8266Client-";      clientId += String(random(0xffff), HEX);      // Attempt to connect      if (client.connect(clientId.c\_str())) {        Serial.println("connected");        // Once connected, publish an announcement...        // client.publish("DATN20172605/Device", "MQTT Server is Connected");        // ... and resubscribe        client.subscribe("DATN20172605/warning");      } else {        Serial.print("failed, rc=");        Serial.print(client.state());        Serial.println(" try again in 5 seconds");        // Wait 5 seconds before retrying        delay(5000);      }    }  }  void setup() {    Serial.begin(9600);    setup\_wifi();    client.setServer(mqtt\_server, 1883);    client.setCallback(callback);    pinMode(12,OUTPUT);    dht.begin();  }  void loop() {    // Connect MQTT    if (!client.connected()) {      reconnect();    }    // Wait 5 seconds    delay(5000);      // Read MQ6    float g = digitalRead(5);    float ag = analogRead(A0);    // Read DHT11    float h = dht.readHumidity();    float t = dht.readTemperature(false);    if (isnan(h) || isnan(t) || isnan(g) || isnan(ag)) {      Serial.println("Failed to read!");    }    else {      if (g != 0) {        warning();      }      dtostrf(h, 5, 2, humidityTemp);      dtostrf(t, 6, 2, celsiusTemp);      dtostrf(g, 5, 2, gasTemp);      dtostrf(ag, 5, 2, agasTemp);      strcat(data, "NhaCuaHuy01");      strcat(data, "~");      strcat(data, celsiusTemp);      strcat(data, "~");      strcat(data, humidityTemp);      strcat(data, "~");      strcat(data, agasTemp);      client.publish("DATN20172605/Device", data);      strcpy(data, "");    }    client.loop();  } |



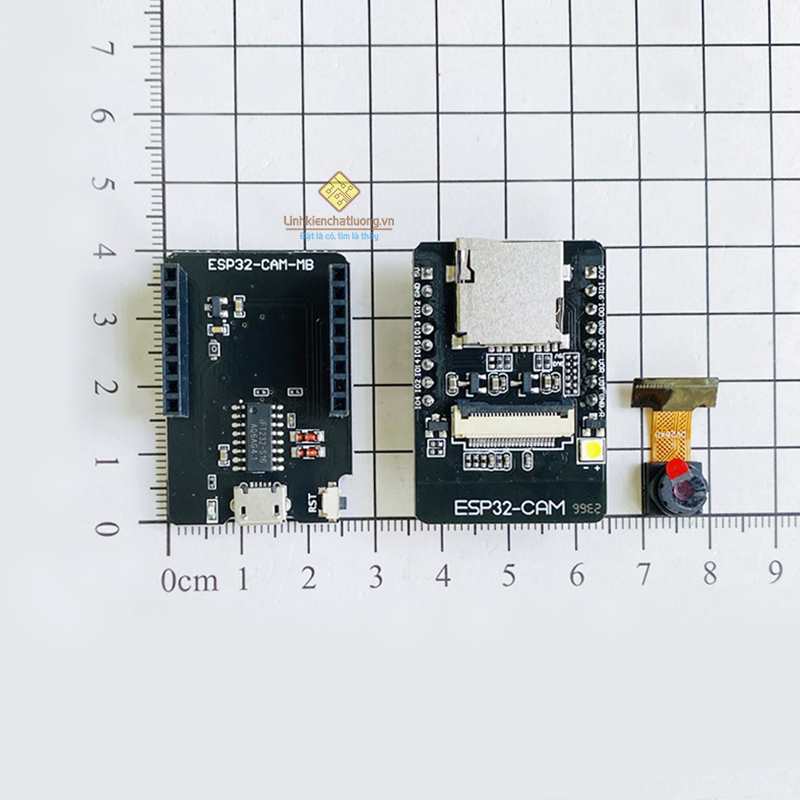
Hình . Sơ đồ kết nối ESP8266 và cảm biến DHT11



Hình . Cảm biến khí gas MQ6

Mã nguồn cho ESP32-CAMERA chụp lại ảnh, gửi về server qua giao thức MQTT để xử lý:

|  |
| --- |
| #include <WiFi.h>  #include <PubSubClient.h>  #include "soc/soc.h"  #include "soc/rtc\_cntl\_reg.h"  #include "esp\_camera.h"  #include "Base64.h"  const char\* ssid = "Huy";  const char\* password = "h150201H";  const char\* mqtt\_server = "broker.emqx.io";  const unsigned int mqtt\_port = 1883;  #define MQTT\_USER               ""  #define MQTT\_PASSWORD           ""  #define MQTT\_PUBLISH\_TOPIC    "DATN20172605/handleImage"  #define MQTT\_SUBSCRIBE\_TOPIC    "DATN20172605/getImage"    WiFiClient espClient;  PubSubClient client(espClient);  #define PWDN\_GPIO\_NUM     32  #define RESET\_GPIO\_NUM    -1  #define XCLK\_GPIO\_NUM      0  #define SIOD\_GPIO\_NUM     26  #define SIOC\_GPIO\_NUM     27  #define Y9\_GPIO\_NUM       35  #define Y8\_GPIO\_NUM       34  #define Y7\_GPIO\_NUM       39  #define Y6\_GPIO\_NUM       36  #define Y5\_GPIO\_NUM       21  #define Y4\_GPIO\_NUM       19  #define Y3\_GPIO\_NUM       18  #define Y2\_GPIO\_NUM        5  #define VSYNC\_GPIO\_NUM    25  #define HREF\_GPIO\_NUM     23  #define PCLK\_GPIO\_NUM     22  String command="";  String cmd="";  String P1="";  String P2="";  String P3="";  String P4="";  String P5="";  String P6="";  String P7="";  String P8="";  String P9="";  byte receiveState=0;  byte cmdState=1;  byte pState=1;  byte questionState=0;  byte equalState=0;  byte semicolonState=0;  void setup() {    WRITE\_PERI\_REG(RTC\_CNTL\_BROWN\_OUT\_REG, 0);      Serial.begin(115200);    randomSeed(micros());    initCamera();    initWiFi();    client.setServer(mqtt\_server, mqtt\_port);    client.setCallback(callback);    client.setBufferSize(2048);  }  void loop() {    if (!client.connected()) {      reconnect();    }    client.loop();  }  void callback(char\* topic, byte\* payload, unsigned int length) {    sendImage();  }  void reconnect() {    // Loop until we're reconnected    while (!client.connected()) {      Serial.print("Attempting MQTT connection...");      // Create a random client ID      String clientId = "ESP32-";      clientId += String(random(0xffff), HEX);      // Attempt to connect      if (client.connect(clientId.c\_str(), MQTT\_USER, MQTT\_PASSWORD)) {        Serial.println("connected");        // Once connected, publish an announcement...        //client.publish(MQTT\_PUBLISH\_TOPIC, "hello world");        // ... and resubscribe        client.subscribe(MQTT\_SUBSCRIBE\_TOPIC);      } else {        Serial.print("failed, rc=");        Serial.print(client.state());        Serial.println(" try again in 5 seconds");        // Wait 5 seconds before retrying        delay(5000);      }    }  }  void initCamera() {    camera\_config\_t config;    config.ledc\_channel = LEDC\_CHANNEL\_0;    config.ledc\_timer = LEDC\_TIMER\_0;    config.pin\_d0 = Y2\_GPIO\_NUM;    config.pin\_d1 = Y3\_GPIO\_NUM;    config.pin\_d2 = Y4\_GPIO\_NUM;    config.pin\_d3 = Y5\_GPIO\_NUM;    config.pin\_d4 = Y6\_GPIO\_NUM;    config.pin\_d5 = Y7\_GPIO\_NUM;    config.pin\_d6 = Y8\_GPIO\_NUM;    config.pin\_d7 = Y9\_GPIO\_NUM;    config.pin\_xclk = XCLK\_GPIO\_NUM;    config.pin\_pclk = PCLK\_GPIO\_NUM;    config.pin\_vsync = VSYNC\_GPIO\_NUM;    config.pin\_href = HREF\_GPIO\_NUM;    config.pin\_sscb\_sda = SIOD\_GPIO\_NUM;    config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;    config.pin\_pwdn = PWDN\_GPIO\_NUM;    config.pin\_reset = RESET\_GPIO\_NUM;    config.xclk\_freq\_hz = 20000000;    config.pixel\_format = PIXFORMAT\_JPEG;      if(psramFound()){      config.frame\_size = FRAMESIZE\_UXGA;      config.jpeg\_quality = 10;      config.fb\_count = 2;    } else {      config.frame\_size = FRAMESIZE\_SVGA;      config.jpeg\_quality = 12;      config.fb\_count = 1;    }    esp\_err\_t err = esp\_camera\_init(&config);    if (err != ESP\_OK) {      Serial.printf("Camera init failed with error 0x%x", err);      ESP.restart();    }    sensor\_t \* s = esp\_camera\_sensor\_get();    if (s->id.PID == OV3660\_PID) {      s->set\_vflip(s, 1);      s->set\_brightness(s, 1);      s->set\_saturation(s, -2);    }    s->set\_framesize(s, FRAMESIZE\_QVGA);    ledcAttachPin(4, 4);    ledcSetup(4, 5000, 8);  }  void initWiFi() {    WiFi.mode(WIFI\_STA);      for (int i=0;i<2;i++) {      WiFi.begin(ssid, password);        delay(1000);      Serial.println("");      Serial.print("Connecting to ");      Serial.println(ssid);        long int StartTime=millis();      while (WiFi.status() != WL\_CONNECTED) {          delay(500);          if ((StartTime+5000) < millis()) break;      }        if (WiFi.status() == WL\_CONNECTED) {        Serial.println("");        Serial.println("STAIP address: ");        Serial.println(WiFi.localIP());        Serial.println("");          pinMode(2, OUTPUT);        for (int j=0;j<5;j++) {          digitalWrite(2,HIGH);          delay(100);          digitalWrite(2,LOW);          delay(100);        }        break;      }    }    if (WiFi.status() != WL\_CONNECTED) {    //若連線失敗      pinMode(2, OUTPUT);      for (int k=0;k<2;k++) {        digitalWrite(2,HIGH);        delay(1000);        digitalWrite(2,LOW);        delay(1000);      }    }  }  String sendImage() {    camera\_fb\_t \* fb = NULL;    fb = esp\_camera\_fb\_get();    if(!fb) {      return "Camera capture failed";    }    char \*input = (char \*)fb->buf;    char output[base64\_enc\_len(3)];    //String imageFile = "data:image/jpeg;base64,";    String imageFile = "";    for (int i=0;i<fb->len;i++) {      base64\_encode(output, (input++), 3);      if (i%3==0) imageFile += String(output);    }    int fbLen = imageFile.length();      String clientId = "ESP32-";    clientId += String(random(0xffff), HEX);    if (client.connect(clientId.c\_str(), MQTT\_USER, MQTT\_PASSWORD)) {        client.beginPublish(MQTT\_PUBLISH\_TOPIC, fbLen, true);      String str = "";      for (size\_t n=0;n<fbLen;n=n+2048) {        if (n+2048<fbLen) {          str = imageFile.substring(n, n+2048);          client.write((uint8\_t\*)str.c\_str(), 2048);        }        else if (fbLen%2048>0) {          size\_t remainder = fbLen%2048;          str = imageFile.substring(n, n+remainder);          client.write((uint8\_t\*)str.c\_str(), remainder);        }      }        client.endPublish();        esp\_camera\_fb\_return(fb);        return "";    }    esp\_camera\_fb\_return(fb);    return "failed, rc="+client.state();  } |



Hình . Module ESP32-CAM Wifi Bluetooth OV2640 kèm đế

### Triển khai mã nguồn xử lý ảnh với python

|  |
| --- |
| import random  import base64  import cv2  import numpy as np  from io import BytesIO  from PIL import Image  import time  from paho.mqtt import client as mqtt\_client  broker = 'broker.emqx.io'  port = 1883  topic = "DATN20172605/handleImage"  publishTopic = "DATN20172605/getImage"  warningTopic = "DATN20172605/warning"  # Generate a Client ID with the subscribe prefix.  client\_id = f'subscribe-{random.randint(0, 100)}'  # username = 'emqx'  # password = 'public'  def connect\_mqtt() -> mqtt\_client:      def on\_connect(client, userdata, flags, rc):          if rc == 0:              print("Connected to MQTT Broker!")          else:              print("Failed to connect, return code %d\n", rc)      client = mqtt\_client.Client(client\_id)      # client.username\_pw\_set(username, password)      client.on\_connect = on\_connect      client.connect(broker, port)      return client  def subscribe(client: mqtt\_client):      def on\_message(client, userdata, msg):          try:              Fire\_Reported = 0              img\_string = msg.payload.decode()              img\_data = base64.b64decode(img\_string)              img\_arr = np.frombuffer(img\_data, dtype=np.uint8)              frame = cv2.imdecode(img\_arr, flags=cv2.IMREAD\_COLOR)              frame = cv2.resize(frame, (960, 540))              blur = cv2.GaussianBlur(frame, (21, 21), 0)              hsv = cv2.cvtColor(blur, cv2.COLOR\_BGR2HSV)              lower = [18, 50, 50]              upper = [35, 255, 255]              lower = np.array(lower, dtype="uint8")              upper = np.array(upper, dtype="uint8")              mask = cv2.inRange(hsv, lower, upper)              no\_red = cv2.countNonZero(mask)              if int(no\_red) > 15000:                  Fire\_Reported = Fire\_Reported + 1              print(Fire\_Reported)              if Fire\_Reported == 1:                  client.publish(warningTopic, "")          except:              print("An exception occurred")      client.subscribe(topic)      client.on\_message = on\_message  def run():      client = connect\_mqtt()      subscribe(client)      client.loop\_start()      while True:          client.publish(publishTopic, "get")          time.sleep(10)  if \_\_name\_\_ == '\_\_main\_\_':      run() |

## Giao diện của một số chức năng đã thực hiện

### Giao diện đăng nhập

A screenshot of a computer

Description automatically generated

Hình . Giao diện đăng nhập

### Giao diện danh sách căn hộ

A screenshot of a computer

Description automatically generated

Hình . Giao diện danh sách căn hộ

### Giao diện thêm mới căn hộ

A screenshot of a computer

Description automatically generated

Hình . Giao diện thêm mới căn hộ

### Giao diện chi tiết căn hộ và xem chỉ số căn hộ

A screenshot of a computer

Description automatically generated

Hình . Giao diện chi tiết căn hộ và xem chỉ số căn hộ

### Giao diện danh sách thiết bị

A screenshot of a computer

Description automatically generated

Hình . Giao diện danh sách thiết bị

### Giao diện thêm mới thiết bị

A screenshot of a computer

Description automatically generated

Hình . Giao diện thêm mới thiết bị

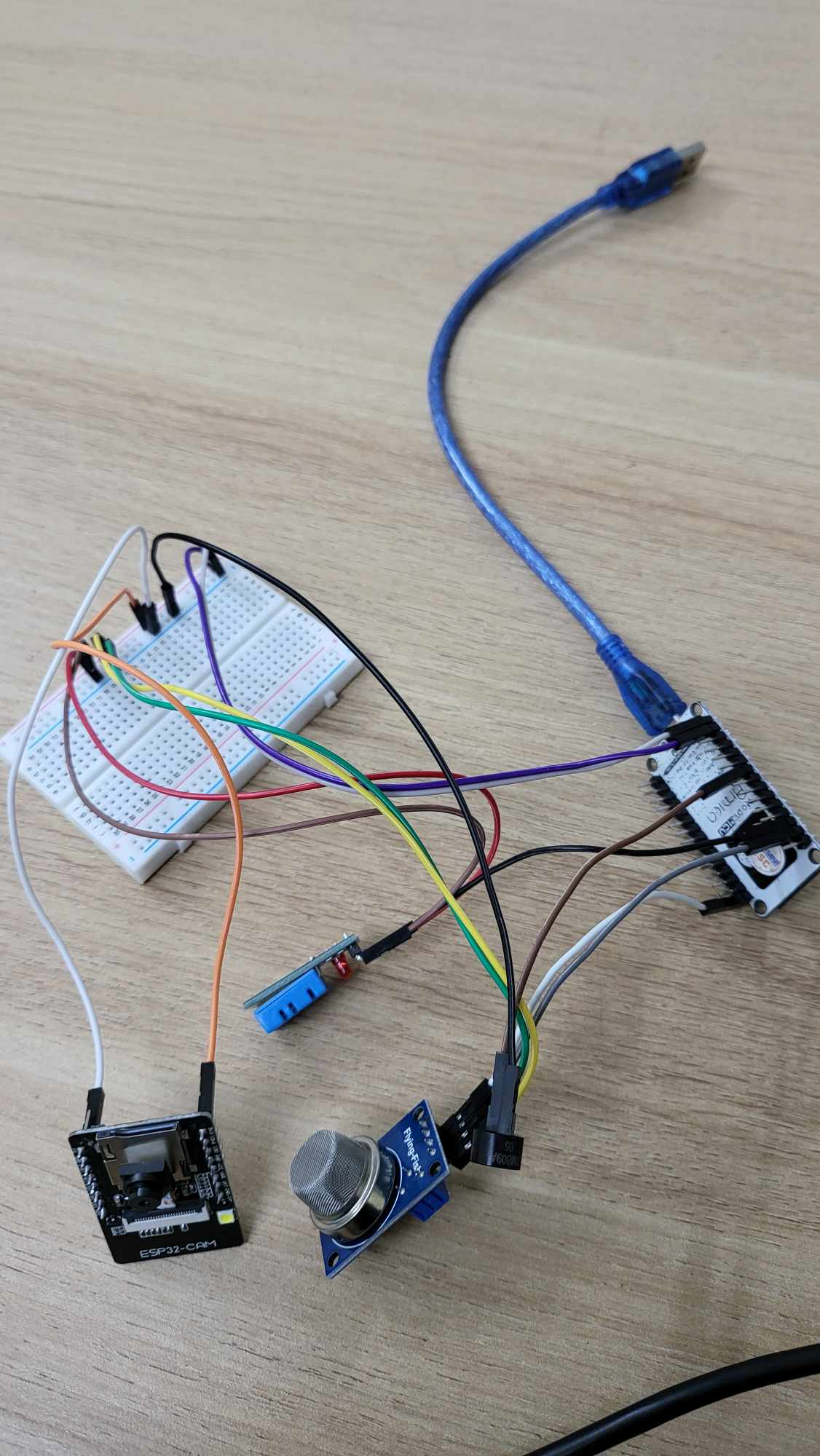
### Giao diện chi tiết thiết bị

A screenshot of a computer

Description automatically generated

Hình . Giao diện chi tiết thiết bị

### Lắp các linh kiện trên board test



Hình . Triển khai lắp linh kiện trên board test

## Kết luận

Như vậy, chương ba đã trình bày về việc triển khai ứng dụng cảnh báo cháy, tiến hành thực hiện một số chức năng cơ bản của ứng dụng cảnh báo cháy.

# KẾT LUẬN

## Kết luận chung

Về kiến thức bản thân:

* Hiểu biết nhiều hơn về ngôn ngữ lập trình C#. Nắm được các kỹ thuật trong lập trình giao thức MQTT và Web API.
* Bổ sung kiến thức về cơ sở dữ liệu, mạng máy tính, cách truyền và nhận dữ liệu từ Client đến Server và các công nghệ mới.
* Nâng cao khả năng tự học, tự tìm hiểu.
* Việc triển khai đồ án tốt nghiệp giúp tác giả có cái nhìn tổng quan về quá trình xây dựng toàn bộ hệ thống, từ việc phân tích, thiết kế hệ thống, đến quá trình triển khai, xây dựng hệ thống, hiển rõ hơn về bản chất, cấu trúc làm việc của một hệ thống.

Về chương trình

* Chương trình đã đáp ứng được các chức năng cơ bản của một ứng dụng cảnh báo cháy cơ bản, giúp cảnh báo nếu có nguy cơ cháy nổ một cách kịp thời.

Tuy nhiên, do thời gian phát triển ứng dụng, cũng như những hạn chế về mặt kiến thức nên đồ án này vẫn còn nhiều thiếu sót, rất mong sự góp ý của thầy cô để đồ án hoàn thiện tốt hơn, đáp ứng nhu cầu của người dùng nhiều hơn.

## Hướng phát triển

Trong tương lai, việc hoàn thiện các chức năng cơ bản và phát triển thêm các chức năng khác của hệ thống là khá dễ dàng đó là nhờ vào việc thiết kế hệ thống theo hướng hướng đối tượng.

Ứng dụng hiện chạy trên nền tảng web. Tuy nhiên, việc phát triển ứng dụng này trên nền tảng di dộng hay ứng dụng desktop là việc dễ dàng nhờ sử dụng Web API.

Cải thiện tư duy hệ thống hiệu quả hơn, kĩ năng code tối ưu hơn, khả năng lập trình hướng đối tượng nói riêng và các ngôn ngữ lập trình hướng đối tượng khác nói chung được nâng cao.

# TÀI LIỆU THAM KHẢO

1. Borade Samar Sarjerao and Amara Prakasarao, “Smart Healthcare Monitoring System Using MQTTProtocol”, in *2018 3rd International Conference for Convergence in Technology (I2CT),* Apr 2018.
2. [Online]. Available: <https://viblo.asia/p/mqtt-la-gi-vai-tro-cua-mqtt-trong-iot-V3m5WL3bKO7>, truy cập cuối cùng ngày 10/12/2021
3. [Online]. Available: <https://viblo.asia/p/nhung-diem-khac-biet-giua-sql-va-nosql-gGJ59b4rKX2>, truy cập cuối cùng ngày 10/12/2021
4. [Online]. Available: https://viblo.asia/p/mongodb-la-gi-co-so-du-lieu-phi-quan-he-bJzKmgoPl9N, truy cập cuối cùng ngày 10/12/2021
5. [Online]. Available: <https://vietnix.vn/tim-hieu-mo-hinh-mvc-la-gi/>, truy cập cuối cùng ngày 14/12/2021
6. [Online]. Available: <http://bugnetproject.com/visual-studio-la-gi/>, truy cập cuối cùng ngày 10/12/2021
7. [Online]. Available: https://viblo.asia/p/tim-hieu-ve-cach-thiet-ke-class-diagram-L4x5xLyY5BM, truy cập cuối cùng ngày 29/12/2021
8. [Online]. Available: [Activity diagram là gì? Cách xây dựng Activity Diagram – Viết luận văn thuê (vietluanvanthue.com)](https://vietluanvanthue.com/activity-diagram-la-gi-cach-xay-dung-activity-diagram/), truy cập cuối cùng ngày 29/12/2021