**ỦY BAN NHÂN DÂN TỈNH BÌNH DƯƠNG**

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**KHOA KỸ THUẬT - CÔNG NGHỆ**

****

**BÁO CÁO MÔN HỌC**

**Xây dựng website bán điện thoại Mobile Store tích hợp hệ thống gợi ý sản phẩm**

**Giáo viên hướng dẫn : ThS. Trần Văn Hữu**

**Nhóm sinh viên thực hiện:**

**Nguyễn Minh Thiện (1524801030068)**

**Nguyễn Văn Khởi (1524801030056)**

LỜI CẢM ƠN

Để hoàn thành tốt đề tài tốt nghiệp này trước hết em xin gửi đến quý thầy, cô giáo trong khoa Kỹ Thuật - Công Nghệ trường Đại Học Thủ Dầu Một lời cảm ơn chân thành.

Đặc biệt, xin gửi đến thầy Nguyễn Tấn Lộc, người đã tận tình hướng dẫn, giúp đỡ trong việc nâng cao kiến thức chuyên môn, phát huy cho người thực hiện tính tự học, tự tìm hiểu, từ đó hình thành cho em phương pháp nghiên cứu, tìm tòi, tiếp cận với khoa học công nghệ.

Vì kiến thức bản thân còn hạn chế, nên trong quá trình xây dựng đề tài cuối kỳ có một số thiếu sót nhất định.

Xin chân thành cảm ơn!

MỤC LỤC

Phần A Giới thiệu

LỜI CẢM ƠN i

MỤC LỤC ii

DANH MỤC HÌNH v

DANH MỤC BẢNG v

Phần B Nội dung

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN VỀ ĐỀ TÀI 1

1.1. Tên đề tài 1

1.2. Lý do chọn đề tài 1

1.3. Mục tiêu của đề tài 2

1.4. Đối tượng, phạm vi nghiên cứu, cách tiếp cận và phương pháp nghiên cứu 2

1.4.1. Đối tượng 2

1.4.2. Phạm vi nghiên cứu 2

1.4.3. Cách tiếp cận 2

1.4.4. Phương pháp nghiên cứu 3

CHƯƠNG 2: GIẢI PHÁP CÔNG NGHỆ 4

2.1. Công nghệ và nền tảng 4

2.2. Tổng quan về NodeJS, Angular 6 4

2.2.1. NodeJS là gì? Đặc điểm của NodeJS 4

2.2.2. Tổng quan về Angular 6 6

2.3. Tổng quan về nền tảng Android 11

2.3.1. Tổng quan 11

2.3.2. Lịch sử 11

2.4. Tổng quan về hệ thống điện toán đám mây Google Firebase 13

2.4.1. Firebase Authentication 13

2.4.2. Firebase Realtime Database 15

2.4.3. Firebase Storage 16

CHƯƠNG 3: XÁC ĐỊNH YÊU CẦU VÀ XÂY DỰNG CHƯƠNG TRÌNH 18

3.1. Yêu cầu tổng thể 18

3.1.1. Quản lý danh mục 18

3.1.2. Quản lý mã QR 18

3.1.3. Thống kê tra cứu mã QR 18

3.2. Yêu cầu chức năng ứng dụng 18

3.2.1. Chức năng dành cho trang quản lý 18

3.2.2. Chức năng dành cho ứng dụng trên điện thoại 18

3.3. Thiết kế Usecase 19

3.3.1. Danh sách Actor 19

3.3.2. Biểu đồ hệ thống 19

3.3.3. Danh sách Usecase 20

3.3.4. Biểu đồ Usecase 21

3.3.5. Đặc tả Usecase 21

3.3.6. Biểu đồ tuần tự các chức năng hệ thống 29

3.4. Thiết kế cơ sở dữ liệu 36

3.4.1. Lưu dữ liệu phòng học 36

3.4.2. Lưu dữ liệu hình ảnh phòng học 36

3.4.3. Lưu lịch sử chỉnh sửa 37

CHƯƠNG 4: GIAO DIỆN CHƯƠNG TRÌNH 38

4.1. Giao diện trang quản lý 38

4.1.1. Giao diện đăng nhập 38

4.1.2. Giao diện Home 38

4.1.3. Giao diện Thông tin cá nhân 38

4.1.4. Giao diện Quản lý mã QR 39

4.1.5. Giao diện Thêm mã QR 39

4.1.6. Giao diện Danh sách mã QR 40

4.1.7. Giao diện Lịch sử 40

4.1.8. Giao diện Chi tiết mã QR 41

4.2. Giao diện ứng dụng trên điện thoại 41

4.2.1. Giao diện Đăng nhập 41

4.2.2. Giao diện Màn hình chính 42

4.2.3. Giao diện Camera quét mã 43

4.2.4. Giao diện Hiển thị nội dung kết quả sau khi quét 43

CHƯƠNG 5: KẾT LUẬN 44

5.1. Ưu điểm của đề tài 44

5.2. Nhược điểm của đề tài 44

5.3. Hướng phát triển của đề tài 44

Phần C Tài liệu tham khảo

TÀI LIỆU THAM KHẢO 45

DANH MỤC HÌNH

[Hình 1. Logo của NodeJS 4](#_Toc5415750)

[Hình 2. Logo của Angular 6 6](#_Toc5415751)

[Hình 3. Logo của Android 11](#_Toc5415752)

[Hình 4. Logo của Firebase 13](#_Toc5415753)

[Hình 5. Biểu đồ hệ thống 19](#_Toc5415754)

[Hình 6. Biểu đồ Use case 21](#_Toc5415755)

[Hình 7. Biểu đồ tuần tự Use case đăng nhập 29](#_Toc5415756)

[Hình 8. Biểu đồ tuần tự Use case Đăng xuất 30](#_Toc5415757)

[Hình 9. Biểu đồ tuần tự Use case Thêm mã QR 30](#_Toc5415758)

[Hình 10. Biểu đồ tuần tự Use case Sửa 31](#_Toc5415759)

[Hình 11. Biểu đồ tuần tự Use case Xóa 31](#_Toc5415760)

[Hình 12. Biểu đồ tuần tự Use case Quản lý mã QR 32](#_Toc5415761)

[Hình 13. Biểu đồ tuần tự Use case Quét mã QR 32](#_Toc5415762)

[Hình 14. Biểu đồ tuần tự Use case Hiển thị thông tin 33](#_Toc5415763)

[Hình 15. Biểu đồ tuần tự Use case Thông tin cá nhân 33](#_Toc5415764)

[Hình 16. Biểu đồ tuần tự Use case Lịch sử 34](#_Toc5415765)

[Hình 17. Biểu đồ tuần tự Use case Danh sách mã QR 34](#_Toc5415766)

[Hình 18. Biểu đồ tuần tự Use case Xem chi tiết 35](#_Toc5415767)

[Hình 19. Cơ sở dữ liệu phòng học 36](#_Toc5415768)

[Hình 20. Cơ sở dữ liệu hình ảnh phòng học 36](#_Toc5415769)

[Hình 21. Cơ sở dữ liệu lịch sử chỉnh sửa 37](#_Toc5415770)

[Hình 22. Giao diện đăng nhập của hệ thống quản lý mã QR 38](#_Toc5415771)

[Hình 23. Giao diện Home của hệ thống quản lý mã QR 38](#_Toc5415772)

[Hình 24. Giao diện Thông tin cá nhân của hệ thống quản lý mã QR 38](#_Toc5415773)

[Hình 25. Giao diện Quản lý của hệ thống quản lý mã QR 39](#_Toc5415774)

[Hình 26. Giao diện Thêm mã QR của hệ thống quản lý mã QR 39](#_Toc5415775)

[Hình 27. Giao diện Danh sách mã QR của hệ thống quản lý mã QR 40](#_Toc5415776)

[Hình 28. Giao diện Lịch sử của hệ thống quản lý mã QR 40](#_Toc5415777)

[Hình 29. Giao diện xem chi tiết của hệ thống quản lý mã QR 41](#_Toc5415778)

[Hình 30. Giao diện Đăng nhập trên ứng dụng điện thoại 41](#_Toc5415779)

[Hình 31. Giao diện Đăng nhập trên ứng dụng điện thoại 42](#_Toc5415780)

[Hình 32. Giao diện Màn hình chính trên ứng dụng điện thoại 42](#_Toc5415781)

[Hình 33. Giao diện Camera quét mã trên điện thoại 43](#_Toc5415782)

[Hình 34. Giao diện Hiển thị nội dung kết quả sau khi quét trên ứng dụng điện thoại 43](#_Toc5415783)

DANH MỤC BẢNG

[Bảng 1. Danh sách Actor 19](#_Toc5302903)

[Bảng 2. Danh sách Use case 20](#_Toc5302904)

1. GIỚI THIỆU TỔNG QUAN VỀ ĐỀ TÀI
   1. Tên đề tài

“Xây dựng website bán điện thoại Mobile Store tích hợp hệ thống gợi ý sản phẩm”

* 1. Lý do chọn đề tài

Hiện tại nhu cầu mua các sản phẩm điện tử thông qua các website bán hàng ngày nay trở nên phổ biến. Đây là một trong những cách tiếp cận với khách hàng nhanh hơn. Vì thế cần phải có giải pháp thông minh để đưa ra giải pháp marketing phù hợp với khách hàng thông qua website.

Hệ thống gợi ý là một trong những thành phần trong hệ thống thông tin. Mục đích của nó là hỗ trợ người dùng thực hiện gợi ý những nội dung phù hợp với sở thích, quan điểm, hành động, hành vi của người khác, của chính bản thân đối tượng cần gợi ý hoặc dựa trên những đặc điểm của các sản phẩm cần gợi ý.

Hệ thống gợi ý ngày một quan trọng để giúp marketing sản phẩm một cách thuận tiện cũng như đưa sản phẩm đến đúng đối tượng hơn.

Đề tài sẽ gồm 2 thành phần: Xây dựng wesbite bán điện thoại và xây dựng tính năng gợi ý sản phẩm

* 1. Mục tiêu của đề tài

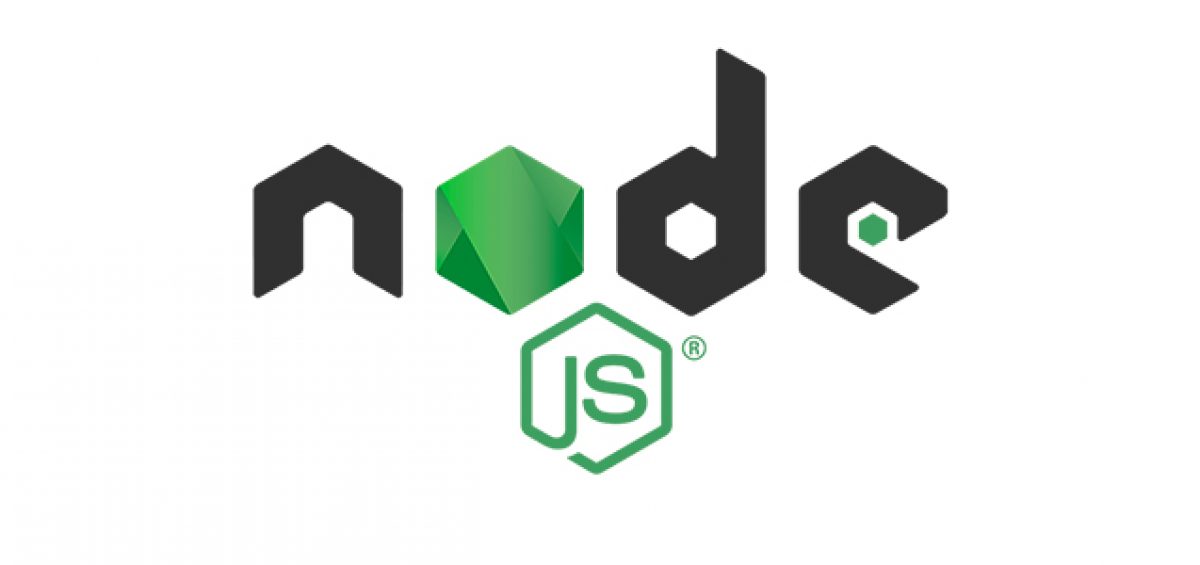
Mục tiêu chính của đề tài là nghiên cứu, xây dựng một hệ thống gợi ý dựa trên hành vi của người dùng để

* Quảng bá sản phẩm đến đúng đối tượng cần quảng bá
* Người dùng chỉ cần truy cập vào website và hệ thống gợi ý sẽ gợi ý các sản phẩm một cách nhanh chóng, độ trễ thấp.
* Ưu thế của việc sử dụng hệ thống gợi ý là hoàn toàn tự động và không cần sự can thiệp của con người.
  1. Đối tượng, phạm vi nghiên cứu, cách tiếp cận và phương pháp nghiên cứu
     1. Đối tượng
* Người tiêu dùng có nhu cầu mua điện thoại.
* Cửa hàng bán điện thoại Mobile Store.
  + 1. Phạm vi nghiên cứu
* Thu thập, phân tích dữ liệu ở của hàng mobile store.
  + 1. Cách tiếp cận
* Tìm hiểu thực trạng việc marking hiện tại của cửa hàng đối với website.
* Thu thập ý kiến từ chủ cửa hàng và nghiên cứu thị hiếu mua hàng của người tiêu dùng.

1. GIẢI PHÁP CÔNG NGHỆ
   1. Công nghệ, thuật toán được sử dụng

Hệ thống gợi ý được xây dựng trên nền tảng Java

* Sử dụng Spring Framework để xây dựng tính năng gợi ý
* Thuật toán
  1. Tổng quan về NodeJS, Angular 6
     1. NodeJS là gì? Đặc điểm của NodeJS



1. Logo của NodeJS

* NodeJs là một nền tảng được phát triển độc lập được xây dựng trên *javascript* *runtime* của *chrome’s*, với NodeJs chúng ta có thể xây dựng được ứng dụng mạng nhanh chóng và dễ dàng mở rộng. Nodejs được xây dựng và phát triển từ năm 2009 và được bảo trợ từ công ty *Joyent*. Khả năng đáp ứng của NodeJs là rất nhanh bởi NodeJs được viết hầu hết bởi ngôn ngữ *C.*
* NodeJs là một ngôn ngữ *Server*. Nghĩa là NodeJs có đủ khả năng để tạo mọi trang *web* theo nhu cầu. Giống như nhiều ngôn ngữ *Server* khác, các page được tạo bằng NodeJs cũng sử dụng *Html*, *Css*, *Javascript* còn NodeJs đóng vai trò là code điều khiển.
* Ưu điểm:
* Cú pháp ngắn gọn, dễ nhớ.
* Xử lý không đồng bộ tốt
* Khả năng xử lý song song tuyệt vời
* Có nhiều *packages* hỗ trợ để tạo nhiều chức năng cho *site*, các *packages* được cài đặt bằng câu lệnh giống như cài đặt *gem* trong *ruby on rails*
* Hỗ trợ nhiều *template engine* để *render view*
* Có nhiều *framework*. Được sử dụng phổ biến nhất là: *ExpressJs*, *SailsJS*
* Khả năng sử dụng lại code: dễ dàng tạo ra các *module*, *helper*, *library* để sử dụng ở nhiều nơi
* Hỗ trợ *security* tốt. NodeJs mặc định hỗ trợ chống tấn công *Cross*-*site* *script*
* Nhược điểm:
* Chưa phân tách rõ ràng 3 phần *model*, *view*, *controller* theo mô hình MVC, ngay cả khi sử dụng *ExpressJs* *framework* thì vẫn phải thủ công cấu trúc theo MVC
* Việc truy xuất dữ liệu thông qua nhiều bảng gặp nhiều hạn chế.
* Việc custom url tương đối khó khăn.
* Khi *include* một *module*, hay require một *class* vào một nơi nào đó để dùng cần phải khai báo đường dẫn tuyệt đối, hoặc tương đối tới *file* chứa class, *module* đó. Điều này khá bất tiện nếu độ sâu của *file* chứa *class*, *module* lớn
* Khai báo *routing* thủ công, nghĩa là muốn một *url* nào đó có hiệu lực cần phải khai báo thủ công *routing* cho nó, không có chế độ *auto*

* + 1. Tổng quan về Angular 6



1. Logo của Angular 6

* *Angular* là một *framework* phát triển trên nền *Javascript* của *Google* dùng để xây dựng những ứng dụng phía client sử dụng *HTML*, *CSS* và một ngôn ngữ lập trình như *Javascript* hoặc *Typescript*.
* Tại sao nên chọn *Angular* mà không phải chọn một *Javascript* *Framework* khác ?
* *Angular* làm cho *HTML* của chúng ta trở nên linh hoạt hơn.
* *Angular* làm cho code *HTML* của chúng ta trở nên mạnh mẽ hơn với những đặc trưng như điều kiện “*if*”, vòng lặp “*for*” và những biến địa phương “*local variables*”.
* *Angular* có cơ chế *binding data* mạnh mẽ, chúng ta có thể dễ dàng hiển thị các *field* từ *data model* của chúng ta, theo dõi những thay đổi và cập nhật lại người dùng.
* *Angular* hoạt động theo thiết kế *module*. Ứng dụng của chúng ta sẽ xây được xây dựng từ những khối *module* độc lập, làm cho việc xây dựng sẽ dễ dàng và có thể tái sử dụng được nội dung.
* *Angular* hỗ trợ việc giao tiếp với những *back-end service*. Điều này sẽ dễ dàng cho việc tích hợp những *back-end service* để việc giải quyết các bài toán *logic* nhanh chóng.
* *Angular* cực kì phổ biến với hàng triệu nhà phát triển đang sử dụng nó.
* *Angular* được *google* phát triển và cập nhật qua từng năm. Phiên bản càng mới thì hiệu năng càng cao và hỗ trợ càng nhiều trình duyệt phổ biến hiện nay.
* Vậy trong *Angular* có gì?

*Angular* - ứng dụng là một tập hợp những *component* (thành phần) và một vài *services* (dịch vụ) cung cấp chức năng trên những *component* đó.

**Angular Application = Component + Component … + services Component = Template + Class + Metadata**

* Mỗi *component* bao gồm một mẫu *HTML* cái mà sẽ *render* ra một “*frame*” giao diện người dùng.
* Một *class* để *code* những gì liên kết với *view.* *Class* chứa được những thuộc tính, những phần tử dữ liệu có sẵn để phục vụ cho các *view* và các phương thức thực hiện hành động cho *view*, chẳng hạn như sự phản hồi của một nút bấm.
* Một *component* cũng có *metadata* cung cấp thêm những thông tin của *component* cho *Angular*. *Metadata* này để xác định *class* là một *Angular* *component*.
* Vậy, một *component* sẽ hình thành từ một *view* xác định từ một mẫu *HTML*, *Code* xử lý sẽ được định nghĩa với một *class* và thông tin bổ xung cho *class* thì được xác định bằng *metadata*.
* *Angular* có tất cả bao nhiêu phiên bản ? Các phiên bản của *Angular* lần lượt ra đời theo thứ tự sau đây : *AngularJS*, *Angular* *2*, *Angular* *4*, *Angular 6.* Hiện tại *Angular 6* là phiên bản mới nhất của *Angular*.. Theo đánh giá của giới lập trình viên trên thế giới thì so với một số đặc tính mới bổ sung cao cấp khác. *Angular 6* bổ sung một số tính năng mới, nâng cao hiệu năng, cũng như xử lý các lỗi xuất hiện trên các phiên bản trước (*Angular 4, Angular 2*).

1. *Cải tiến hiệu năng của Angular 6*

* Tối ưu hóa *build* *production* bằng việc sử dụng công cụ *build optimizer* được tích hợp sẵn vào trong *CLI*. Công cụ này tối ưu tree *shark* và loại bỏ *code* dư thừa.
* Cải thiện tốc độ biên dịch bằng việc sử dụng *TypeScript* *transforms*, giờ đây khi build sẽ sử dụng lệnh “*ng serve –aot*”. AOT sẽ cải thiện *performance* *khi load page* và nó được dùng để *deploy app* lên *production*.
* Sử dụng *Static Injector* thay thế *Reflective -Injector*: Do *injector* không cần *intl and Reflect polyfills* nên sẽ làm giảm kích thước của ứng dụng.
* Có thêm option mới “*preserveWhitespaces*” nhằm loại bỏ các khoảng trắng không cần thiết trong *template* giúp giảm kích thước của *bundle* khi biên dịch. Chức nằng này off làm mặc định, để thay đổi chỉ cần thêm dòng config trong file *tsconfig.json* hoặc có thể thêm *config* trực tiếp vào *component*

1. *Tính năng mới*

* Chú trọng vào đơn giản hoá việc xây dựng *PWA*, để ứng dụng có thể chạy trên trình duyệt.
* Tính năng tối ưu hoá làm cho ứng dụng nhẹ hơn bằng cách loại bỏ những dòng *code* không cần thiết.
* Làm những thành phần Thiết kế Tài nguyên được sánh với kết xuất đồ hoạ dựa trên *server*.

*PWA*, sản phẩm của nỗ lực hợp tác giữa *Google* và *Mozilla*, là khái niệm về cho phép phát triển ứng dụng cho trình duyệt cung cấp những trải nghiệm chất lượng cao và muợt mà. Hỗ trợ *PWA* trong *Angular* yêu cầu rất thành thạo trong việc phát triển ứng dụng ; phiên bản 5 có thể sẽ khiến việc sử dụng dễ dàng hơn.

1. *Cách upgrade lên Angular 6 và việc upgrade đã được đội ngũ phát triển hỗ trợ khiến việc nâng cấp trở nên dễ dàng hơn*

So với các phiên bản trước thì với phiên bản này việc nâng cấp các package đã nhẹ nhàng hơn nhiều, chỉ cần chạy một dòng lệnh là mọi thứ đã được cập nhật.

npm uninstall -g @angular/cli

npm cache verify

Nếu phiên bản npm <5 thì sử dụng *npm cache clean*

npm install -g @angular/cli@next

Thay đổi các phiên bản gói góc trong tệp *package.json* thành ^6.0.0-rc.5

"dependencies": {

"@angular/animations": "^6.0.0-rc.5",

"@angular/cdk": "^6.0.0-rc.12",

"@angular/common": "^6.0.0-rc.5",

"@angular/compiler": "^6.0.0-rc.5",

"@angular/core": "^6.0.0-rc.5",

"@angular/forms": "^6.0.0-rc.5",

"@angular/http": "^6.0.0-rc.5",

"@angular/material": "^6.0.0-rc.12",

"@angular/platform-browser": "^6.0.0-rc.5",

"@angular/platform-browser-dynamic": "^6.0.0-rc.5",

"@angular/router": "^6.0.0-rc.5",

"core-js": "^2.5.5",

"karma-jasmine": "^1.1.1",

"rxjs": "^6.0.0-uncanny-rc.7",

"rxjs-compat": "^6.0.0-uncanny-rc.7",

"zone.js": "^0.8.26"

},

"devDependencies": {

"@angular-devkit/build-angular": "~0.5.0",

"@angular/cli": "^6.0.0-rc.5",

"@angular/compiler-cli": "^6.0.0-rc.5",

"@types/jasmine": "2.5.38",

"@types/node": "~8.9.4",

"codelyzer": "~4.1.0",

"jasmine-core": "~2.5.2",

"jasmine-spec-reporter": "~3.2.0",

"karma": "~1.4.1",

"karma-chrome-launcher": "~2.0.0",

"karma-cli": "~1.0.1",

"karma-coverage-istanbul-reporter": "^0.2.0",

"karma-jasmine": "~1.1.0",

"karma-jasmine-html-reporter": "^0.2.2",

"postcss-loader": "^2.1.4",

"protractor": "~5.1.0",

"ts-node": "~5.0.0",

"tslint": "~5.9.1",

"typescript": "^2.7.2"

}

Tiếp theo cập nhật gói angular angi local sang phiên bản tiếp theo và cài đặt các gói đã đề cập ở trên.

rm -rf node\_modules dist # use rmdir /S/Q node\_modules dist in Windows

Command Prompt; use rm -r -fo node\_modules,dist in Windows PowerShell

npm install --save-dev @angular/cli@next

npm install

Định dạng cấu hình Angular CLI đã được thay đổi từ phiên bản CLI 6.0.0-rc.2 góc cạnh và cấu hình hiện tại của bạn có thể được cập nhật tự động bằng cách chạy lệnh sau. Nó sẽ loại bỏ tập tin cấu hình cũ .angular-cli.json và sẽ viết mới angular.json tập tin.

ng update @angular/cli --migrate-only --from=1.7.4

* 1. Tổng quan về nền tảng Android



1. Logo của Android
   * 1. Tổng quan

* Android là một hệ điều hành dựa trên nền tảng *Linux* được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Tổng công ty Android, với sự hỗ trợ tài chính từ *Google* và sau này được chính *Google* mua lại vào năm 2005. Android ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy Android được bán vào năm 2008.
* Android có mã nguồn mở và *Google* phát hành mã nguồn theo Giấy phép *Apache*. Chính mã nguồn mở cùng với một giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên nhiệt huyết được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị, bằng một loại ngôn ngữ lập trình *Java* có sửa đổi. Vào tháng 10 năm 2012, có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ *Google Play*, cửa hàng ứng dụng chính của Android, ước tính khoảng 25 tỷ lượt.
  + 1. Lịch sử
* Tổng công ty Android (*Android, Inc*.) được thành lập tại *Palo Alto*, *California* vào tháng 10 năm 2003 bởi *Andy Rubin* (đồng sáng lập công ty *Danger*), *Rich Miner* (đồng sáng lập Tổng công ty Viễn thông *Wildfire*), *Nick Sears* (từng là Phó giám đốc *T-Mobile*), và *Chris White* (trưởng thiết kế và giao diện tại *WebTV*) để phát triển, theo lời của *Rubin*, "các thiết bị di động thông minh hơn có thể biết được vị trí và sở thích của người dùng". Dù những người thành lập và nhân viên đều là những người có tiếng tăm, Tổng công ty Android hoạt động một cách âm thầm, chỉ tiết lộ rằng họ đang làm phần mềm dành cho điện thoại di động.
* *Google* mua lại Tổng công ty Android vào ngày 17 tháng 8 năm 2005, biến nó thành một bộ phận trực thuộc *Google*. Những nhân viên của chủ chốt của Tổng công ty Android, gồm *Rubin*, *Miner* và *White*, vẫn tiếp tục ở lại công ty làm việc sau thương vụ này.
* Vào thời điểm đó không có nhiều thông tin về Tổng công ty, nhưng nhiều người đồn đoán rằng *Google* dự tính tham gia thị trường điện thoại di động sau bước đi này.
* Tại *Google*, nhóm do *Rubin* đứng đầu đã phát triển một nền tảng thiết bị di động phát triển trên nền nhân *Linux*. *Google* quảng bá nền tảng này cho các nhà sản xuất điện thoại và các nhà mạng với lời hứa sẽ cung cấp một hệ thống uyển chuyển và có khả năng nâng cấp. *Google* đã liên hệ với hàng loạt hãng phần cứng cũng như đối tác phần mềm, bắn tin cho các nhà mạng rằng họ sẵn sàng hợp tác với các cấp độ khác nhau.
* Ngày càng nhiều suy đoán rằng *Google* sẽ tham gia thị trường điện thoại di động xuất hiện trong tháng 12 năm 2006. Tin tức của *BBC* và Nhật báo phố *Wall* chú thích rằng *Google* muốn đưa công nghệ tìm kiếm và các ứng dụng của họ vào điện thoại di động và họ đang nỗ lực làm việc để thực hiện điều này. Tháng 9 năm 2007, *InformationWeek* đăng tải một nghiên cứu của *Evalueserve* cho biết *Google* đã nộp một số đơn xin cấp bằng sáng chế trong lĩnh vực điện thoại di động.
* Ngày 5 tháng 11 năm 2007, Liên minh thiết bị cầm tay mở (*Open Handset Alliance*), một hiệp hội bao gồm nhiều công ty trong đó có *Texas* *Instruments*, Tập đoàn *Broadcom*, *Google*, *HTC*, *Intel*, *LG*, Tập đoàn *Marvell* *Technology*, *Motorola*, *Nvidia*, *Qualcomm*, *Samsung* *Electronics, Sprint Nextel* và *T-Mobile* được thành lập với mục đích phát triển các tiêu chuẩn mở cho thiết bị di động. Cùng ngày, Android cũng được ra mắt với vai trò là sản phẩm đầu tiên của Liên minh, một nền tảng thiết bị di động được xây dựng trên nhân *Linux* phiên bản 2.6. Chiếc điện thoại chạy Android đầu tiên được bán ra là *HTC Dream*, phát hành ngày 22 tháng 10 năm 2008. Biểu trưng của hệ điều hành Android mới là một con rôbốt màu xanh lá cây do hãng thiết kế *Irina Blok* tại *California* vẽ.
* Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng; ví dụ như phiên bản 1.5 *Cupcake* (bánh bông lan nhỏ có kem) tiếp nối bằng phiên bản 1.6. Phiên bản mới nhất hiện nay là 9.0 *Pie* ra mắt vào tháng 8 năm 2018. Vào năm 2010, *Google* ra mắt loạt thiết bị *Nexus*—một dòng sản phẩm bao gồm điện thoại thông minh và máy tính bảng chạy hệ điều hành Android, do các đối tác phần cứng sản xuất. *HTC* đã hợp tác với *Google* trong chiếc điện thoại thông minh *Nexus* đầu tiên, *Nexus One*. Kể từ đó nhiều thiết bị mới hơn đã gia nhập vào dòng sản phẩm này, như điện thoại *Nexus* 4 và máy tính bảng *Nexus* 10, lần lượt do *LG* và *Samsung* sản xuất. *Google* xem điện thoại và máy tính bảng *Nexus* là những thiết bị Android chủ lực của mình, với những tính năng phần cứng và phần mềm mới nhất của Android.
  1. Tổng quan về hệ thống điện toán đám mây Google Firebase



1. Logo của Firebase
   * 1. Firebase Authentication

* Hầu hết các ứng dụng cần biết danh tính của người dùng. Biết danh tính của người dùng cho phép ứng dụng lưu dữ liệu người dùng trên đám mây một cách an toàn và cung cấp trải nghiệm được cá nhân hóa giống nhau trên tất cả các thiết bị của người dùng.
* Firebase Authentication cung cấp dịch vụ *back-end*, *SDK* dễ sử dụng và thư viện *UI* được tạo sẵn để xác thực người dùng với ứng dụng của bạn. Hỗ trợ xác thực bằng mật khẩu, số điện thoại, nhà cung cấp nhận dạng liên kết phổ biến như *Google*, *Facebook* và *Twitter*, …
* Firebase Authentication tích hợp chặt chẽ với các dịch vụ *Firebase* khác và kết hợp với các tiêu chuẩn công nghiệp như *OAuth 2.0* và *OpenID Connect*, do đó có thể dễ dàng tích hợp với phụ trợ tùy chỉnh.
* Khả năng chính
* *FirebaseUI Auth* cung cấp giải pháp xác thực thả xuống xử lý các luồng *UI* để đăng nhập người dùng bằng địa chỉ *email* và mật khẩu, số điện thoại và với các nhà cung cấp nhận dạng được liên kết phổ biến, bao gồm Đăng nhập *Google* và Đăng nhập *Facebook*.
* *Firebase SDK Authentication*
* *Xác thực dựa trên email và mật khẩu:* Xác thực người dùng bằng địa chỉ *email* và mật khẩu của họ. *Firebase Authentication SDK* cung cấp các phương thức để tạo và quản lý người dùng sử dụng địa chỉ *email* và mật khẩu của họ để đăng nhập. *Firebase Authentication* cũng xử lý gửi *email* đặt lại mật khẩu.
* *Federated identity provider integration:* Xác thực người dùng bằng cách tích hợp với các nhà cung cấp nhận dạng liên kết. *Firebase Authentication SDK* cung cấp các phương thức cho phép người dùng đăng nhập bằng tài khoản *Google*, *Facebook*, *Twitter* và *GitHub* của họ.
* *Xác thực số điện thoại:* Xác thực người dùng bằng cách gửi tin nhắn SMS đến điện thoại của họ.
* *Tích hợp hệ thống xác thực tùy chỉnh:* Kết nối hệ thống đăng nhập hiện có của ứng dụng của bạn với *Firebase Authentication SDK* và có quyền truy cập vào *Firebase Realtime Database* và các dịch vụ Firebase khác.
* *Ẩn danh:* Sử dụng các tính năng yêu cầu xác thực mà không yêu cầu người dùng đăng nhập trước bằng cách tạo tài khoản ẩn danh tạm thời. Nếu người dùng sau đó chọn đăng ký, bạn có thể nâng cấp tài khoản ẩn danh thành tài khoản thông thường, để người dùng có thể tiếp tục nơi họ rời đi.
* Cơ chế hoạt động
* Để đăng nhập người dùng vào ứng dụng của bạn, trước tiên bạn phải có thông tin xác thực từ người dùng. Các thông tin đăng nhập này có thể là địa chỉ *email* và mật khẩu của người dùng hoặc mã thông báo OAuth từ nhà cung cấp nhận dạng được liên kết. Sau đó, bạn chuyển các thông tin đăng nhập này đến *Firebase Authentication SDK*. Các dịch vụ *back-end* của *Firebase* sau đó sẽ xác minh các thông tin đăng nhập đó và trả lời phản hồi cho khách hàng.
* Sau khi đăng nhập thành công, bạn có thể truy cập thông tin hồ sơ cơ bản của người dùng và bạn có thể kiểm soát quyền truy cập của người dùng vào dữ liệu được lưu trữ trong các sản phẩm *Firebase* khác. Bạn cũng có thể sử dụng mã thông báo xác thực được cung cấp để xác minh danh tính của người dùng trong các dịch vụ phụ trợ của riêng bạn.
  + 1. Firebase Realtime Database
* Firebase Realtime Database lưu trữ và đồng bộ dữ liệu dạng *NoSQL* trên *cloud database*. Dữ liệu được đồng bộ hóa trên tất cả các máy trạm trong thời gian thực và vẫn khả dụng khi ứng dụng trong trạng thái ngoại tuyến.
* Firebase Realtime Database là một *cloud-hosted database.* Dữ liệu được lưu trữ dưới dạng *JSON* và được đồng bộ hóa trong thời gian thực cho mọi máy khách được kết nối. Khi xây dựng các ứng dụng đa nền tảng với *SDK iOS, Android* và *JavaScript*, tất cả các khách hàng của bạn sẽ chia sẻ một phiên bản Cơ sở dữ liệu thời gian thực và tự động nhận các bản cập nhật với dữ liệu mới nhất.
* Cơ chế hoạt động
* Firebase Realtime Database cho phép xây dựng các ứng dụng một cách phong phú bằng cách cho phép truy cập an toàn vào cơ sở dữ liệu trực tiếp từ mã phía máy khách. Dữ liệu được duy trì cục bộ và ngay cả khi ngoại tuyến, các sự kiện thời gian thực vẫn tiếp tục phát sinh, mang đến cho người dùng cuối trải nghiệm phản hồi. Khi thiết bị lấy lại kết nối, Cơ sở dữ liệu thời gian thực sẽ đồng bộ hóa các thay đổi dữ liệu cục bộ với các cập nhật từ xa xảy ra trong khi máy khách ngoại tuyến, tự động hợp nhất mọi xung đột.
* *Realtime Database* cung cấp một ngôn ngữ quy tắc dựa trên biểu thức linh hoạt, được gọi là *Firebase Realtime Database Security Rules*, để xác định cách cấu trúc dữ liệu của bạn và khi nào có thể đọc hoặc ghi dữ liệu. Khi được tích hợp với *Firebase Authentication,* nhà phát triển có thể xác định ai có quyền truy cập vào dữ liệu nào và cách họ có thể truy cập dữ liệu đó.
* *Realtime Database* là một cơ sở dữ liệu *NoQuery* và do đó có các tối ưu hóa và chức năng khác nhau so với cơ sở dữ liệu quan hệ. *Realtime Database API* được thiết kế để chỉ cho phép các hoạt động có thể được thực thi nhanh chóng. Điều này cho phép xây dựng trải nghiệm thời gian thực tuyệt vời có thể phục vụ hàng triệu người dùng mà không ảnh hưởng đến khả năng phản ứng. Do đó, điều quan trọng là phải suy nghĩ về cách người dùng cần truy cập dữ liệu của bạn và sau đó cấu trúc dữ liệu phù hợp.
  + 1. Firebase Storage
* Cloud Storage được xây dựng cho các nhà phát triển ứng dụng cần lưu trữ và tương tác nội dung do người dùng tạo, chẳng hạn như ảnh hoặc video.
* Cloud Storage của Firebase là dịch vụ lưu trữ đối tượng mạnh mẽ, đơn giản và tiết kiệm chi phí được xây dựng cho quy mô *Google*. *Firebase SDK* cho *Cloud Storage* thêm bảo mật của *Google* vào tệp tải lên và tải xuống cho các ứng dụng *Firebase*, bất kể chất lượng mạng. Sử dụng *SDK* của *Firebase* để lưu trữ hình ảnh, âm thanh, video hoặc nội dung khác do người dùng tạo. Trên máy chủ, sử dụng *Google* *Cloud* *Storage*, để truy cập cùng các tệp.
* Khả năng chính
* Hoạt động mạnh mẽ: *SDK Firebase* cho *Cloud Storage* thực hiện tải lên và tải xuống bất kể chất lượng mạng. Tải lên và tải xuống mạnh mẽ, có nghĩa là chúng khởi động lại ở nơi chúng dừng lại, tiết kiệm thời gian và băng thông của người dùng của bạn.
* Độ bảo mật cao: *SDK Firebase* cho *Cloud Storage* tích hợp với *Firebase Authentication* để cung cấp xác thực đơn giản và trực quan cho các nhà phát triển. Bạn có thể sử dụng mô hình bảo mật khai báo của chúng tôi để cho phép truy cập dựa trên tên tệp, kích thước, loại nội dung và metadata.
* Khả năng mở rộng cao: *SDK Firebase* cho *Cloud Storage* được xây dựng cho quy mô *exabyte* khi ứng dụng của bạn bị *virus*. Dễ dàng phát triển từ nguyên mẫu đến sản xuất bằng cách sử dụng cùng một cơ sở hạ tầng hỗ trợ *Spotify* và *Google Photos*.
* Cơ chế hoạt động
* Các nhà phát triển sử dụng *SDK Firebase* cho *Cloud Storage* để tải lên và tải xuống các tệp trực tiếp từ máy khách. Nếu kết nối mạng kém, máy khách có thể thử lại thao tác ngay tại nơi nó dừng, tiết kiệm thời gian và băng thông cho người dùng.
* *Cloud Storage* lưu trữ các tệp trong nhóm *Google Cloud Storage*, giúp chúng có thể truy cập được thông qua cả *Firebase* và *Google Cloud*. Điều này cho phép linh hoạt tải lên và tải xuống các tệp từ ứng dụng khách di động thông qua *SDK Firebase* và xử lý phía máy chủ như lọc hình ảnh hoặc chuyển mã video bằng *Google Cloud Platform*. *Cloud Storage* tự động chia tỷ lệ, nghĩa là không cần phải di chuyển sang bất kỳ nhà cung cấp nào khác.
* *SDK Firebase* cho *Cloud Storage* tích hợp hoàn hảo với *Firebase Authentication* để xác định người dùng và *Firebase* cung cấp ngôn ngữ bảo mật khai báo cho phép bạn đặt các điều khiển truy cập trên các tệp hoặc nhóm tệp riêng lẻ để có thể đặt tệp ở chế độ công khai hoặc riêng tư theo ý muốn.

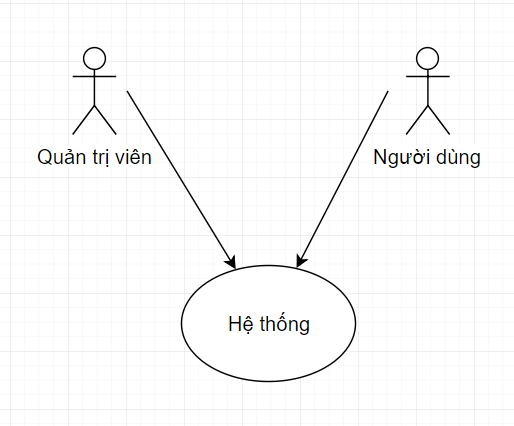
1. XÁC ĐỊNH YÊU CẦU VÀ XÂY DỰNG CHƯƠNG TRÌNH
   1. Yêu cầu tổng thể
      1. Quản lý danh mục

* Danh mục Đăng nhập.
* Danh mục quản lý mã QR.
  + 1. Quản lý mã QR
* Trang thêm mã QR.
* Trang chỉnh sửa mã QR.
* Trang lịch sử chỉnh sửa.
  + 1. Thống kê tra cứu mã QR
* Thống kê kết quả tra cứu mã QR
  1. Yêu cầu chức năng ứng dụng
     1. Chức năng dành cho trang quản lý
* Đăng kí tài khoản bằng email.
* Đăng nhập thông qua email đã đăng kí.
* Thêm, xóa, sửa mã QR.
* Hiển thị lịch sử chỉnh sửa.
* Hiển thị số lượt tra cứu và hiển thị biểu đồ.
  + 1. Chức năng dành cho ứng dụng trên điện thoại
* Đăng nhập thông qua tài khoản Google.
* Tra cứu mã QR bằng camera.
* Hiển thị những nội dung mà người dùng có thể quan tâm.
  1. Thiết kế Usecase
     1. Danh sách Actor

Một actor hay tác nhân ngoài là một vai trò của một hay nhiều người hay vật thể trong sự tương tác với hệ thống.

|  |  |  |
| --- | --- | --- |
| **STT** | **Tác nhân** | **Mô tả** |
| 1 | Quản trị viên | Là người quản trị hệ thống, có toàn quyền trên hệ thống |
| 2 | Người dùng | Là người sử dụng hệ thống, có quyền truy cập vào hệ thống |

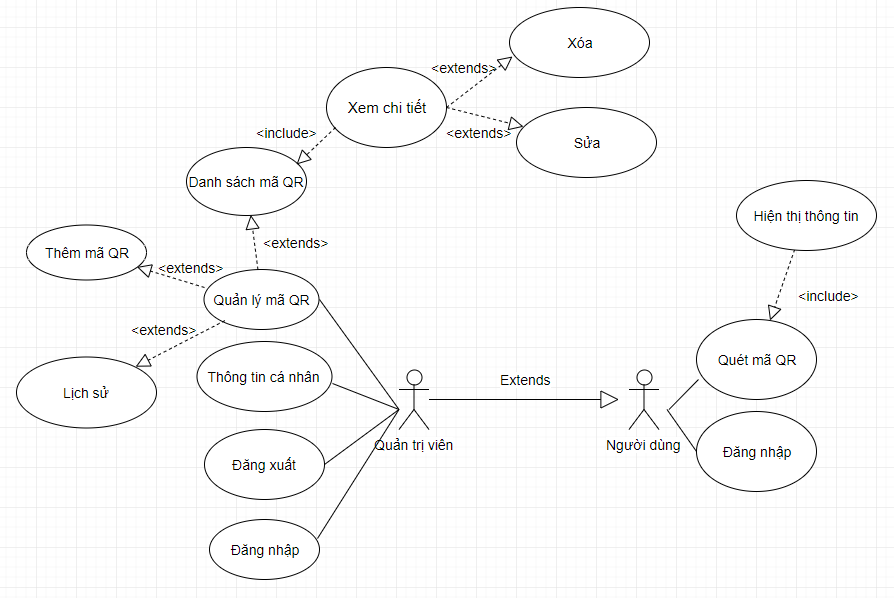
1. Danh sách Actor
   * 1. Biểu đồ hệ thống



1. Biểu đồ hệ thống
   * 1. Danh sách Usecase

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Use case | Ý nghĩa | Actor |
| 1 | Đăng nhập | Cho phép người dùng và quản lý truy cập vào hệ thống | Quản trị viên, Người dùng |
| 2 | Đăng xuất | Cho phép quản lý đăng xuất khỏi hệ thống | Quản trị viên |
| 4 | Quản lý mã QR | Cho phép quản lý xem các thao tác với mã QR | Quản trị viên |
| 5 | Quét mã QR | Cho phép người dùng quét mã QR để truy cập thông tin phòng | Người dùng |
| 6 | Thêm mã QR | Cho phép quản lý thêm thông tin phòng mới | Quản trị viên |
| 7 | Xóa | Cho phép quản lý xóa thông tin phòng đã thêm | Quản trị viên |
| 8 | Sửa | Cho phép quản lý sửa lại thông tin mã QR đã thêm | Quản trị viên |
| 9 | Xem chi tiết | Cho phép quản lý xem chi tiết thông tin mã QR đã thêm | Quản trị viên |
| 10 | Lịch sử | Cho phép quản lý xem lịch sử chỉnh sửa mã QR | Quản trị viên |
| 11 | Danh sách mã QR | Cho phép quản lý xem danh sách mã QR đã thêm | Quản trị viên |
| 12 | Use case Hiển thị thông tin | Cho phép người dùng xem thông tin của mã QR | Người dùng |

1. Danh sách Use case
   * 1. Biểu đồ Usecase



1. Biểu đồ Use case
   * 1. Đặc tả Usecase
        1. Use case Đăng nhập

* Tóm tắt: use case này mô tả cách quản trị viên (người dùng) đăng nhập vào hệ thống.
* Actor: Quản trị viên. (Người dùng)
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên (người dùng) muốn đăng nhập vào hệ thống.

(1) Quản trị viên (người dùng) nhập tên đăng nhập và mật khẩu.

(2) Hệ thống kiểm chứng thông tin đăng nhập, nếu trùng khớp với dữ liệu đã lưu thì cho phép quản trị viên (người dùng) đăng nhập vào hệ thống với giao diện tương ứng.

* + Dòng sự kiện khác: nếu trong dòng sự kiện chính, quản trị viên (người dùng) nhập sai tên đăng nhập, mật khẩu, hệ thống sẽ hiển thị thông báo lỗi. Quản trị viên (người dùng) có thể chọn trở về đầu dòng sự kiện chính hoặc hủy bỏ việc đăng nhập, lúc này use case kết thúc.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở màn hình đăng nhập, các chức năng không được hiển thị.
* Trạng thái hệ thống sau khi thực hiện use case: nếu use case thành công actor có thể sử dụng các chức năng được phân quyền.
* Điểm mở rộng: không có.
  + - 1. Use case Đăng xuất
* Tóm tắt: use case này mô tả cách quản trị viên (người dùng) đăng xuất khỏi hệ thống.
* Actor: Quản trị viên. (Người dùng)
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên (người dùng) muốn đăng xuất khỏi hệ thống.

(1) Quản trị viên (người dùng) click vào nút đăng xuất trên giao diện.

(2) Hệ thống bỏ tài khoản đang đăng nhập và trở về giao diện đăng nhập.

* + Dòng sự kiện khác: không có.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở trong giao diện quản lý.
* Trạng thái hệ thống sau khi thực hiện use case: nếu use case thành công actor sẽ quay trở về giao diện đăng nhập.
* Điểm mở rộng: không có.
  + - 1. Use case Thêm mã QR
* Tóm tắt: use case này mô tả cách người quản trị thêm thông tin phòng mới.
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn thêm thông tin phòng mới.

(1) Người quản trị click vào nút thêm mã QR trong giao diện quản lý mã QR.

(2) Hệ thống hiển thị form yêu cầu quản trị viên nhập thông tin phòng vào.

(3) Quản trị viên nhập thông tin và bấm vào nút Sumit

(4) Hệ thống lưu cơ sở dữ liệu, thông báo lên màn hình và trở về giao diện quản lý mã QR

* + Dòng sự kiện khác:

1. Nếu trong quá trình lưu dữ liệu gặp lỗi hiển thị lỗi thì thông báo lỗi trên màn hình và yêu cầu nhập lại.
2. Không kết nối được cơ sở dữ liệu thông báo lỗi và quay trở về giao diện quản lý mã QR

* Các yêu cầu đăc biệt: Một số thông tin nhập vào form yêu cầu bắt buộc
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở màn hình quản lý thông tin QR.
* Trạng thái hệ thống sau khi thực hiện use case: thông tin được lưu
* Điểm mở rộng: không có.
  + - 1. Use case Sửa
* Tóm tắt: use case này mô tả cách người quản trị sửa thông tin phòng đã nhập
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn sửa thông tin phòng đã nhập.

1. Hệ thống hiện thị form có dữ liệu cần sửa
2. Quản trị viên nhập vào form thông tin cần sửa sau đó bấm nút Sumit
3. Hệ thống kiểm tra thông tin sửa và lưu lại, sau đó thông báo sửa thành công và quay trở lại giao diện danh sách mã QR

Dòng sự kiện khác:

1. Nếu trong quá trình lưu dữ liệu gặp lỗi hiển thị lỗi thì thông báo lỗi trên màn hình và yêu cầu nhập lại.
2. Không kết nối được cơ sở dữ liệu thông báo lỗi và quay trở về giao diện quản lý mã QR

* Các yêu cầu đăc biệt: Một số thông tin nhập vào form yêu cầu bắt buộc
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở màn hình chi tiết thông tin
* Trạng thái hệ thống sau khi thực hiện use case: Thông tin được lưu.
* Điểm mở rộng: không có.
  + - 1. Use case Xóa
* Tóm tắt: use case này mô tả cách người quản trị viên xóa thông tin mã QR
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn xóa một mã QR đã tạo trước.

1. Người quản trị click vào nút xóa ở giao diện chi tiết mã QR.
2. Hệ thống xóa dữ liệu trên cơ sở dữ liệu, thông báo xóa thành công và trở về giao diện danh sách mã QR
   * Dòng sự kiện khác: Không kết nối được cơ sở dữ liệu thông báo lỗi và quay trở về giao diện danh sách mã QR.

* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện chi tiết thông tin.
* Trạng thái hệ thống sau khi thực hiện use case: Thông tin được xóa.
* Điểm mở rộng: không có.
  + - 1. Use case Quản lý mã QR
* Tóm tắt: use case này mô tả cách quản trị viên quản lý thông tin QR
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi một người dùng muốn quản lý mã QR.

(1) Người dùng click vào nút quản lý mã QR.

(2) Hệ thống chuyển sang giao diện quản lý mã QR.

* + Dòng sự kiện khác: không có.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện home.
* Trạng thái hệ thống sau khi thực hiện use case: truy cập giao diện thành công.
* Điểm mở rộng: không có.
  + - 1. Use case Quét mã QR
* Tóm tắt: use case này mô tả cách người dùng quét mã QR từ hình ảnh.
* Actor: Người dùng.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi một người dùng muốn xem thông tin phòng học.

(1) Người dùng click vào nút scan trên màn hình điện thoại và quét hình ảnh mã QR chứa thông tin.

(2) Hệ thống kiểm chứng thông tin và chuyển đến giao diện hiển thị thông tin phòng học.

* + Dòng sự kiện khác: nếu mã QR bị hư không khôi phục được thì thông báo không tìm thấy thông tin.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện scan.
* Trạng thái hệ thống sau khi thực hiện use case: Tải dữ liệu quét được.
* Điểm mở rộng: không có.
  + - 1. Use case Hiển thị thông tin
* Tóm tắt: use case này mô tả cách người dùng xem thông tin từ mã quét được.
* Actor: Người dùng.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi một người dùng vừa quét thành công mã QR.

1. Hệ thống tải dữ liệu thành công.
2. Hiển thị thông tin phòng học trên giao diện cho người dùng
   * Dòng sự kiện khác: Nếu tải dữ liệu không thành công, thông báo lỗi và trở lại giao diện scan.

* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Giao diện quét mã QR.
* Trạng thái hệ thống sau khi thực hiện use case: Hiển thị thông tin phòng học.
* Điểm mở rộng: không có.
  + - 1. Use case Thông tin cá nhân
* Tóm tắt: use case này mô tả cách quản trị viên xem thông tin tài khoản.
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn muốn xem thông tin tài khoản.

1. Quản trị viên click vào nút quản lý thông tin.
2. Hệ thống tải dữ liệu từ database và hiển thị lên màn hình
   * Dòng sự kiện khác: nếu tải được dữ liệu thông báo lỗi và trở về giao diện home.

* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện home
* Trạng thái hệ thống sau khi thực hiện use case: hệ thống tải dữ liệu người dùng về.
* Điểm mở rộng: không có.
  + - 1. Use case Lịch sử
* Tóm tắt: use case này mô tả cách quản trị viên xem lịch sử chỉnh sửa mã QR.
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn xem thông tin chỉnh sữa mã QR đã thực hiện.

(1) Người dùng click vào nút xem lịch sử.

(2) Hệ thống tải dữ liệu từ database về và hiển thị cho người dùng.

* + Dòng sự kiện khác: Không tải được dữ liệu thông báo lỗi và trở về giao diện quản lý mà QR.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện quản lý mã QR.
* Trạng thái hệ thống sau khi thực hiện use case: Tải dữ liệu từ Firebase về và hiển thị.
* Điểm mở rộng: không có.
  + - 1. Use case Danh sách mã QR
* Tóm tắt: use case này mô tả cách quản trị viên xem danh sách mã QR đã thêm.
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn xem danh sách mã QR đã thêm.

(1) Quản trị viên click vào nút danh sách mã QR

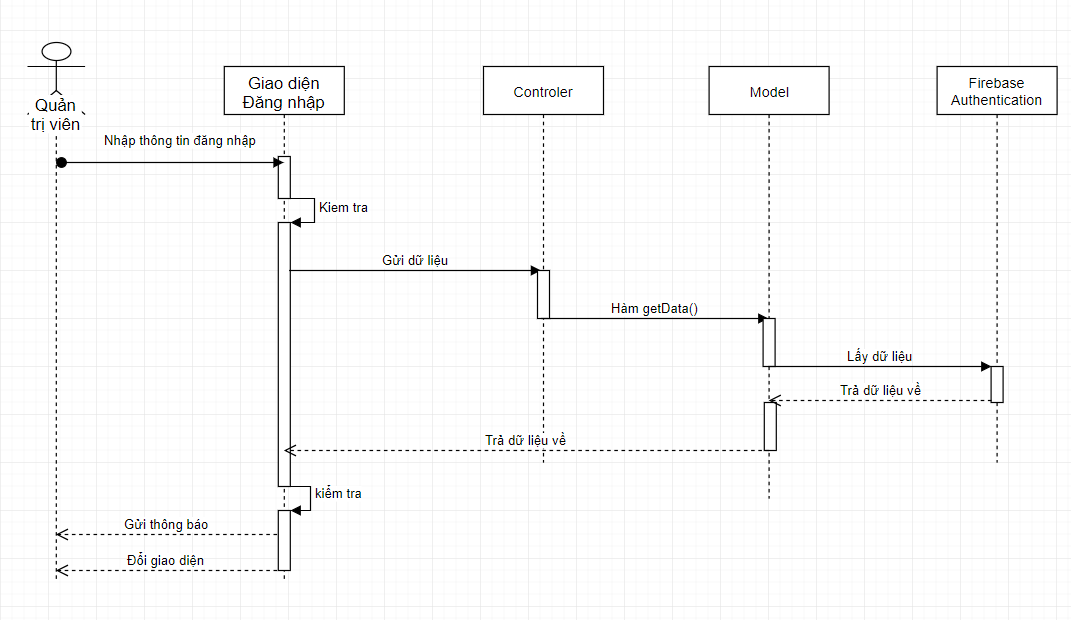
(2) Hệ thống tải dữ liệu từ cơ sở dữ liệu về và hiển thị danh sách.

* + Dòng sự kiện khác: Nếu tải không thành công báo lỗi và kết thúc.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện quản lý mã QR.
* Trạng thái hệ thống sau khi thực hiện use case: Tải dữ liệu từ cơ sở dữ liệu và hiển thị.
* Điểm mở rộng: không có.
  + - 1. Use case Xem chi tiết
* Tóm tắt: use case này mô tả quản trị viên xem chi tiết thông mã QR đã lưu.
* Actor: Quản trị viên.
* Dòng sự kiện:
  + Dòng sự kiện chính: use case này bắt đầu khi quản trị viên muốn xem thông tin mã QR đã lưu.

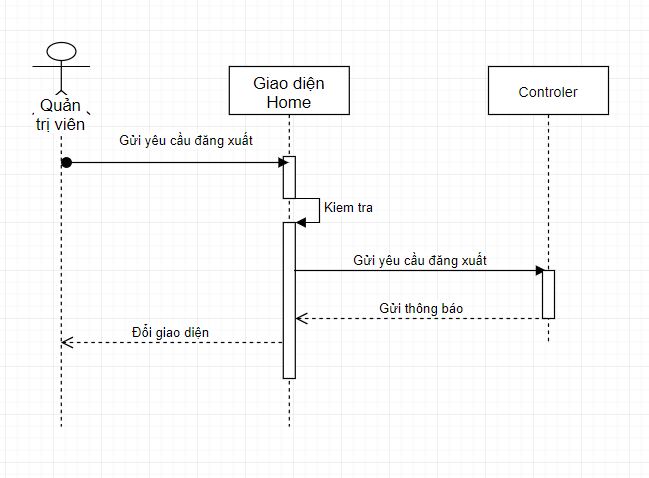
(1) Quản trị viên click vào mã QR trong danh sách.

(2) Hệ thống lấy dữ liệu từ mã QR quản trị viên đã chọn và hiện thị ra màn hình.

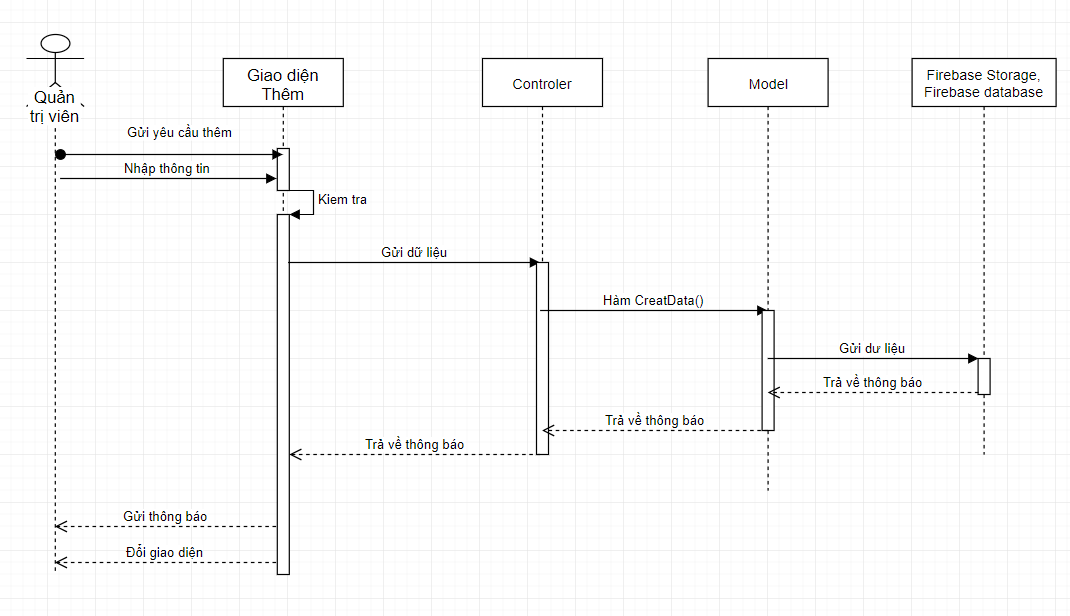
* + Dòng sự kiện khác: Nếu không tải được dữ liệu thông báo lỗi và kết thúc.
* Các yêu cầu đăc biệt: không có.
* Trạng thái hệ thống trước khi thực hiện use case: Đứng ở giao diện danh sách mã QR.
* Trạng thái hệ thống sau khi thực hiện use case: Hệ thống tải dữ liệu mã QR đã chọn và hiện thị ra màn hình.
* Điểm mở rộng: không có.
  + 1. Biểu đồ tuần tự các chức năng hệ thống
       1. Use case Đăng nhập



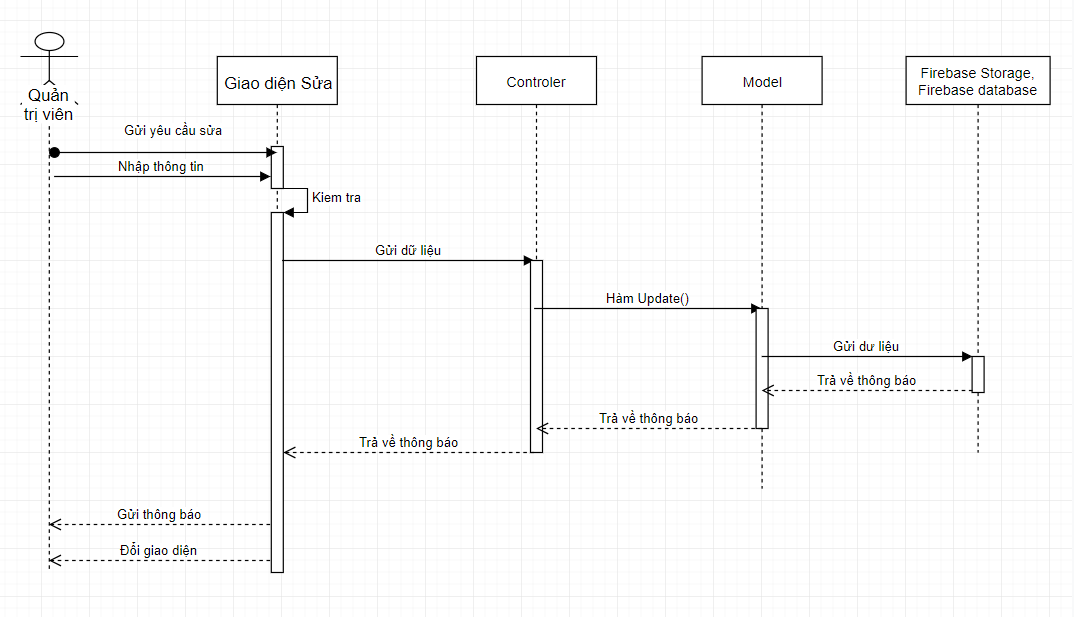
1. Biểu đồ tuần tự Use case đăng nhập
   * + 1. Use case Đăng xuất



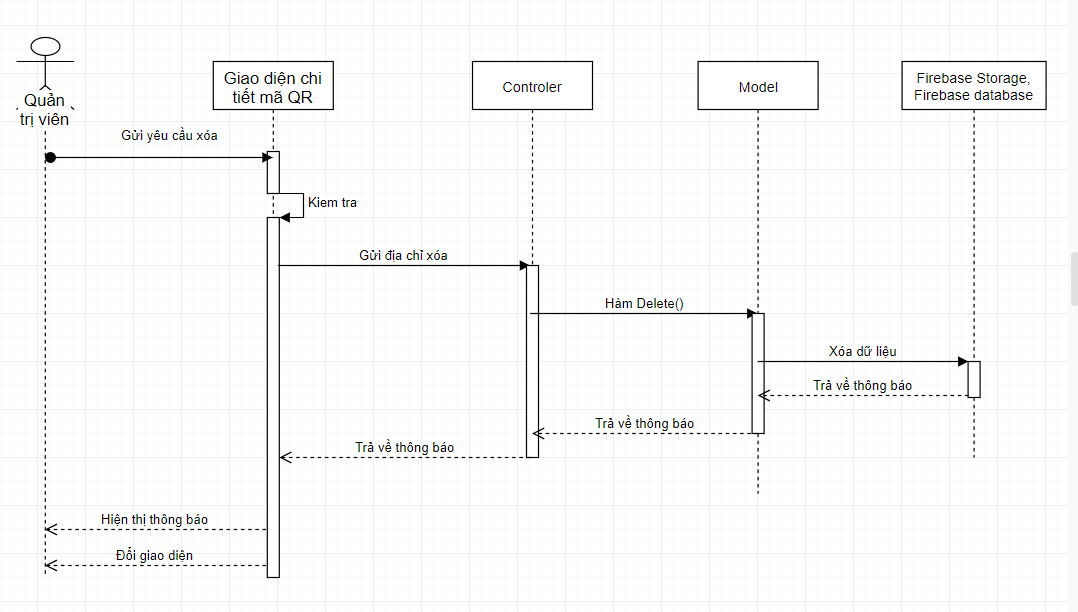
1. Biểu đồ tuần tự Use case Đăng xuất
   * + 1. Use case Thêm mã QR



1. Biểu đồ tuần tự Use case Thêm mã QR
   * + 1. Use case Sửa

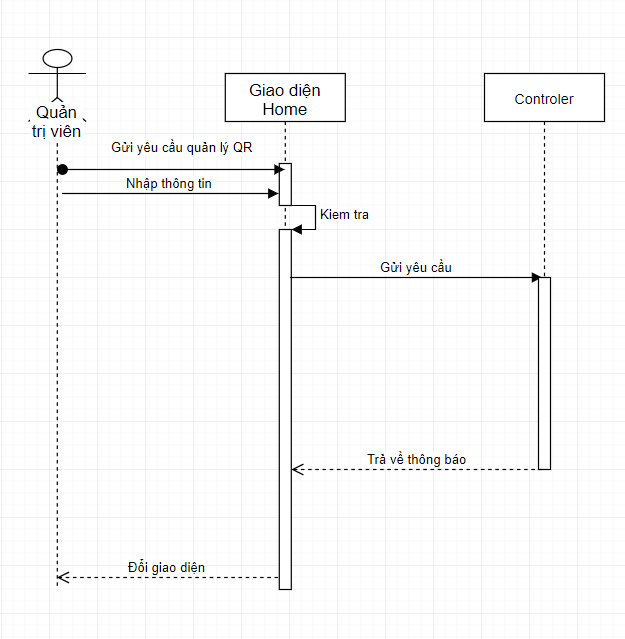


1. Biểu đồ tuần tự Use case Sửa
   * + 1. Use case Xóa

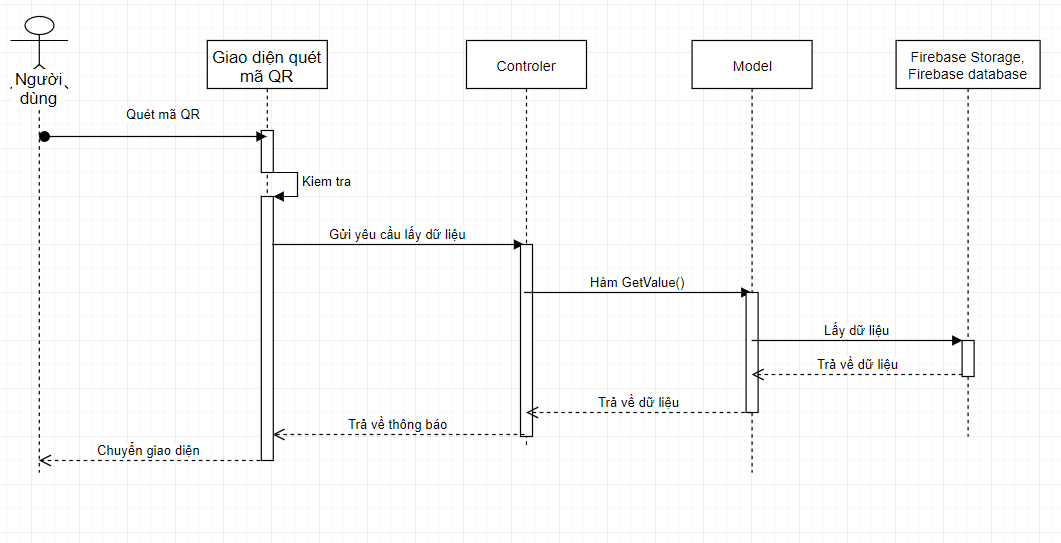


1. Biểu đồ tuần tự Use case Xóa

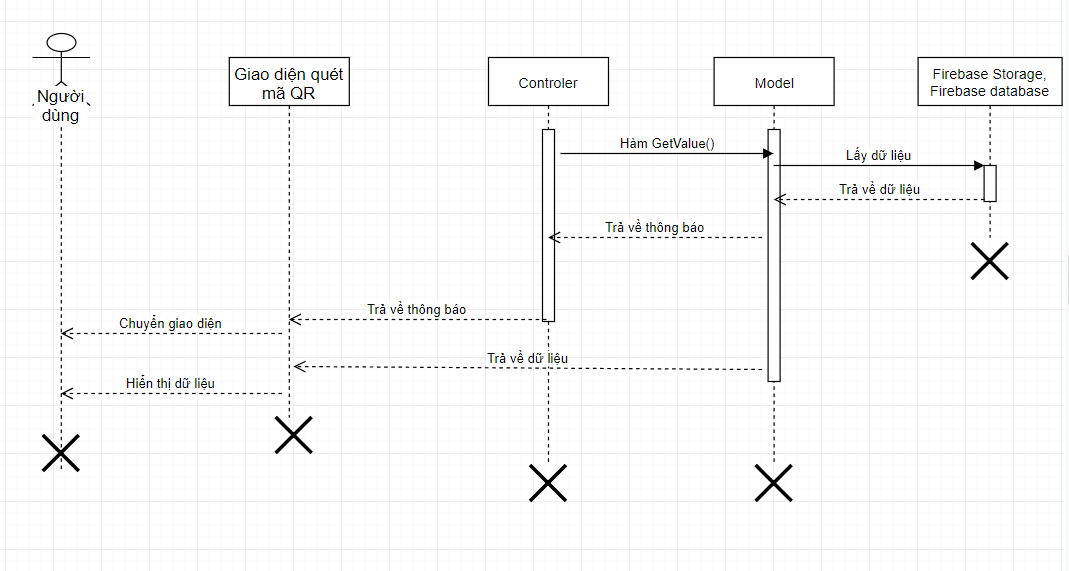
* + - 1. Use case Quản lý mã QR



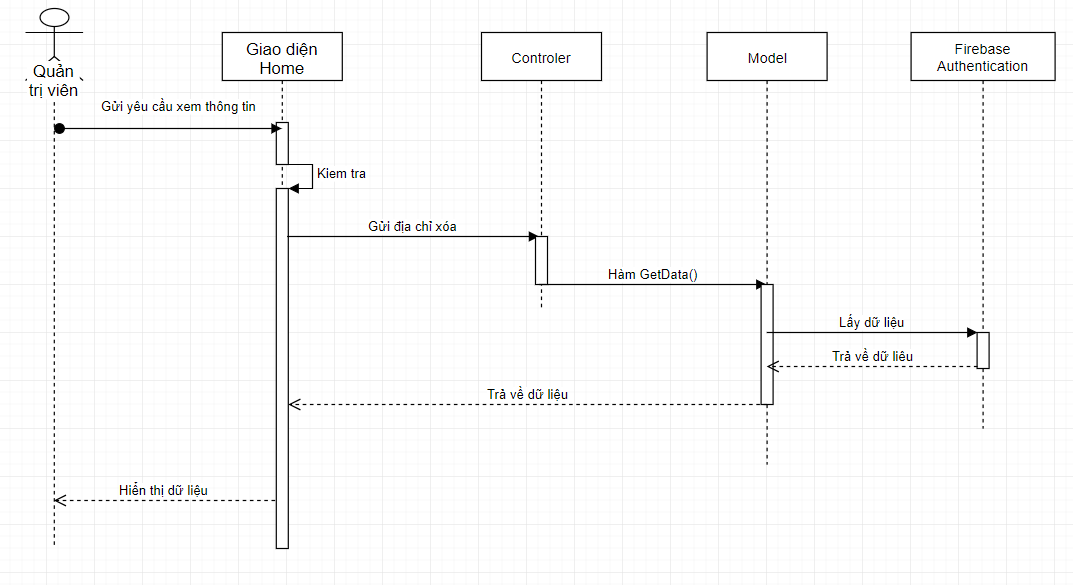
1. Biểu đồ tuần tự Use case Quản lý mã QR
   * + 1. Use case Quét mã QR



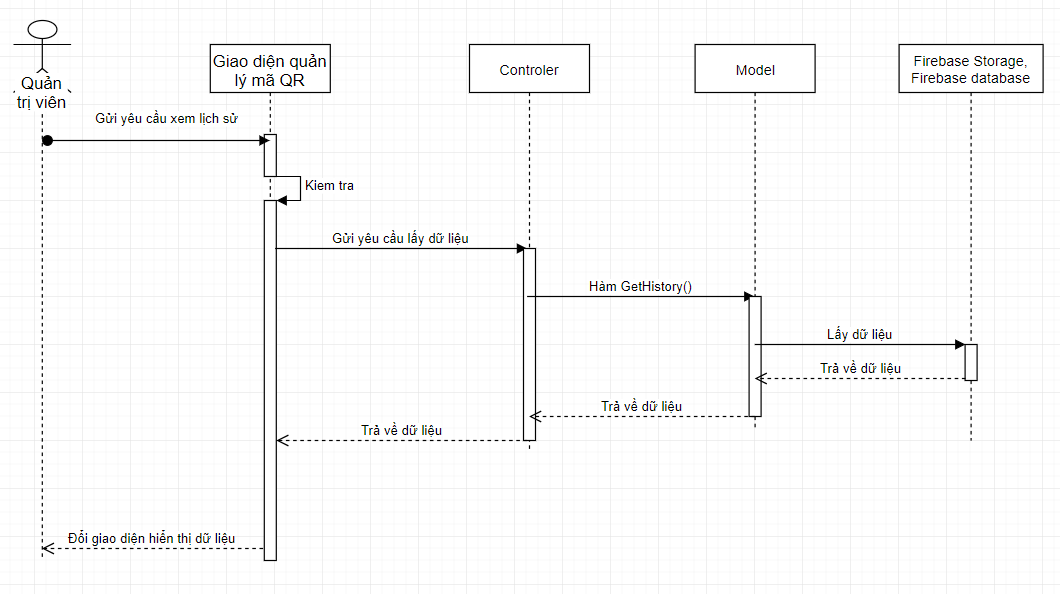
1. Biểu đồ tuần tự Use case Quét mã QR
   * + 1. Use case Hiển thị thông tin



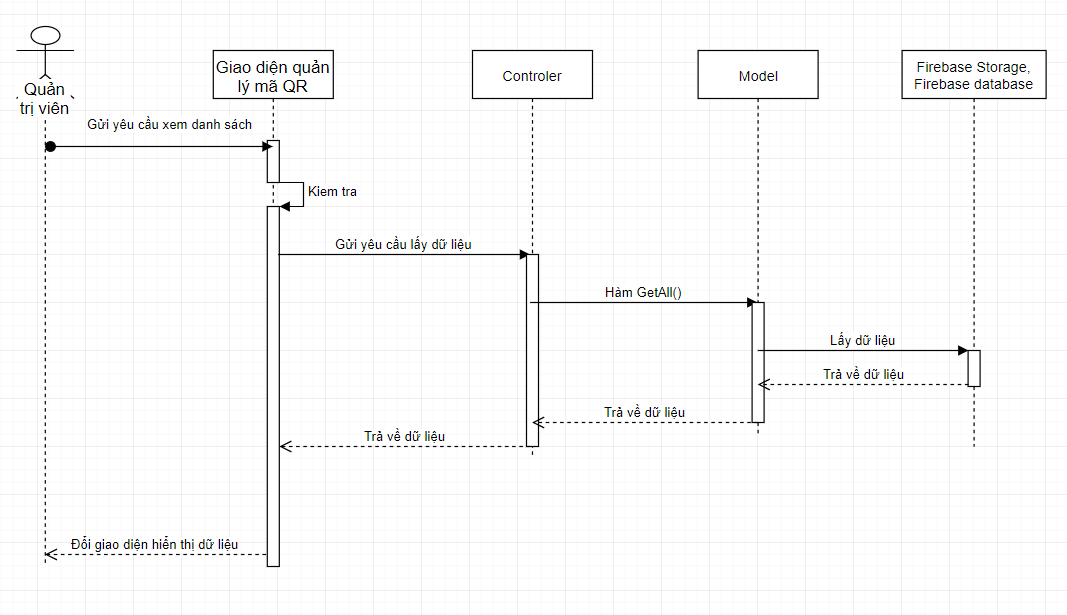
1. Biểu đồ tuần tự Use case Hiển thị thông tin
   * + 1. Use case Thông tin cá nhân



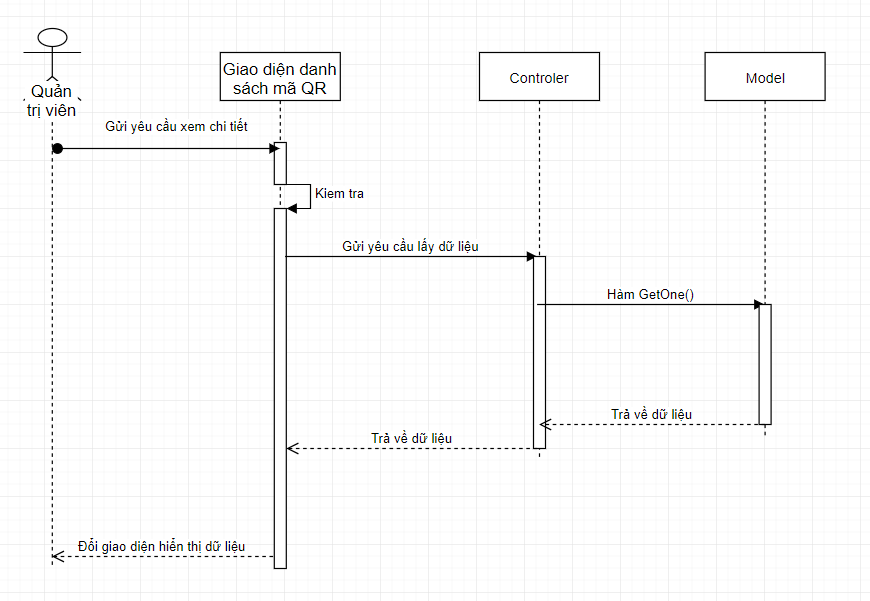
1. Biểu đồ tuần tự Use case Thông tin cá nhân
   * + 1. Use case Lịch sử



1. Biểu đồ tuần tự Use case Lịch sử
   * + 1. Use case Danh sách mã QR

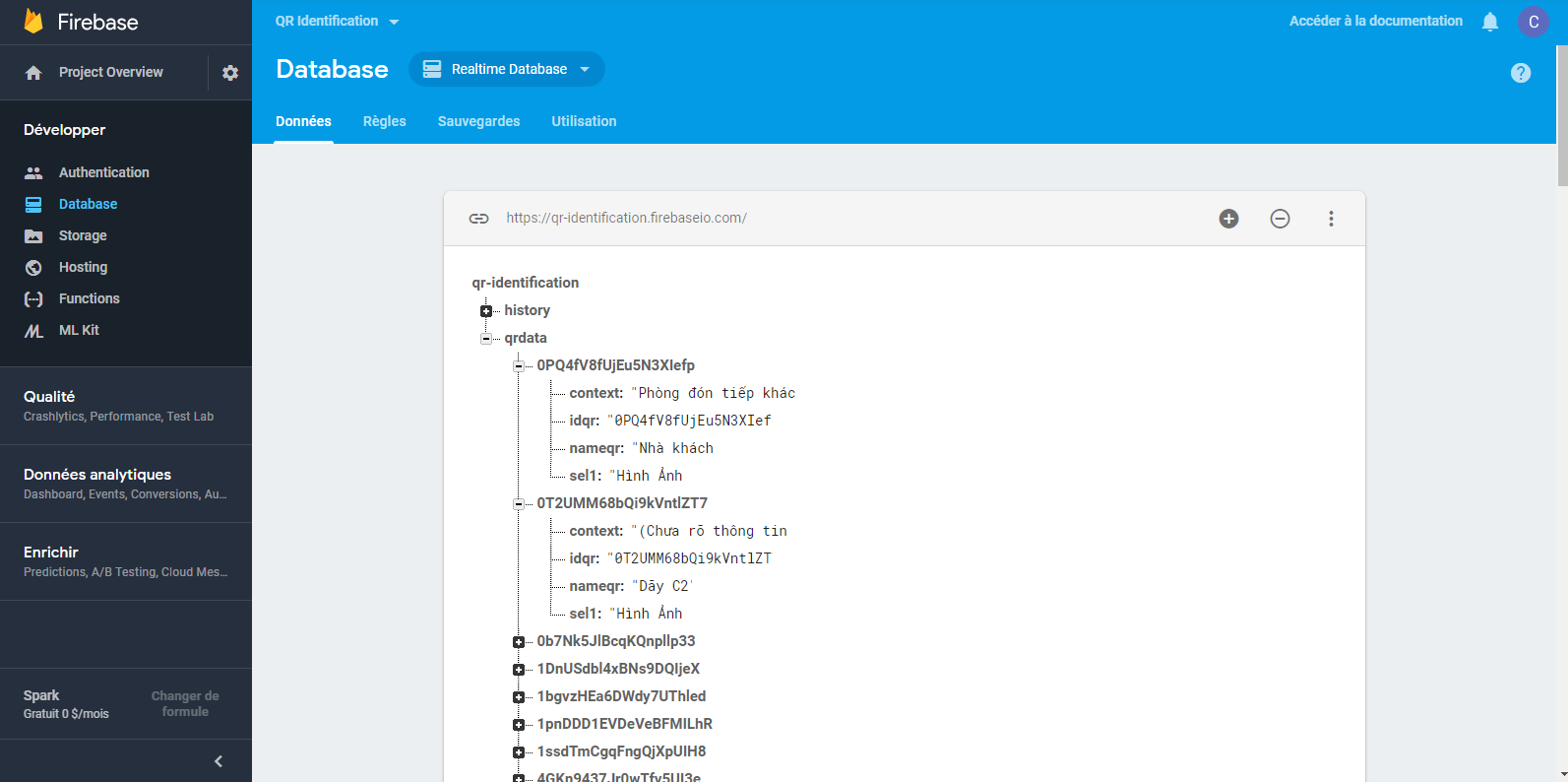


1. Biểu đồ tuần tự Use case Danh sách mã QR
   * + 1. Use case Xem chi tiết

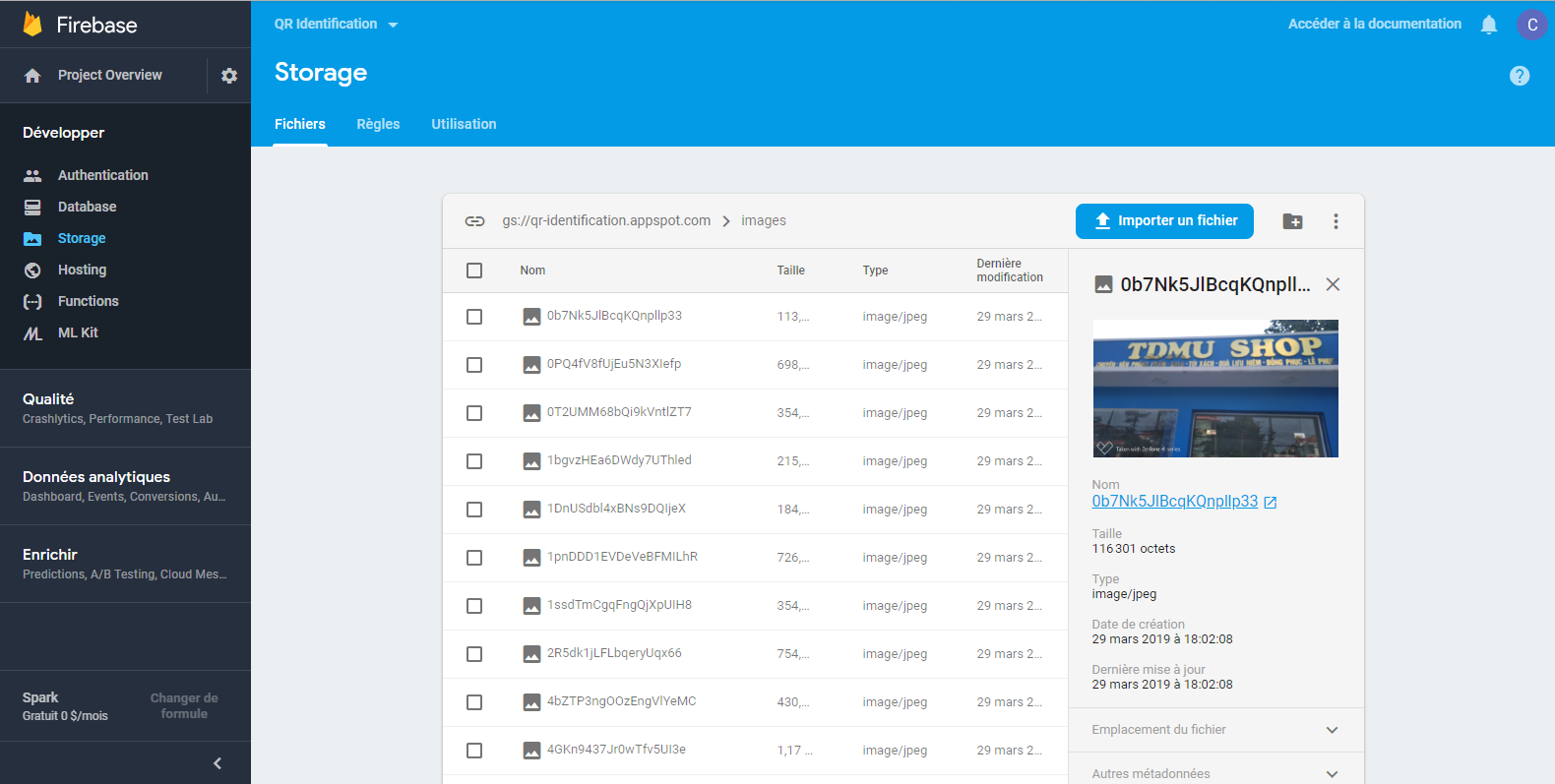


1. Biểu đồ tuần tự Use case Xem chi tiết

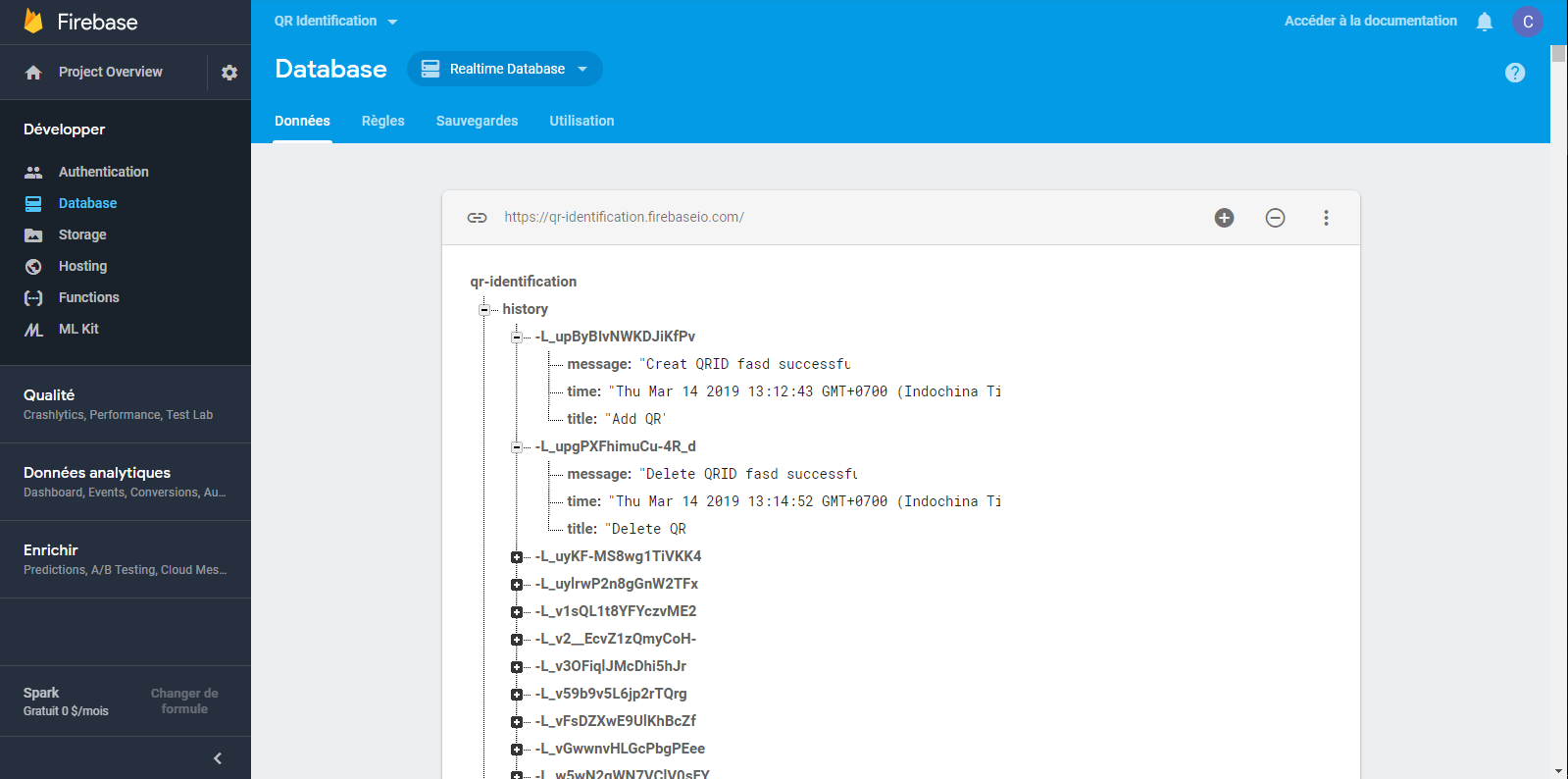
* 1. Thiết kế cơ sở dữ liệu
     1. Lưu dữ liệu phòng học



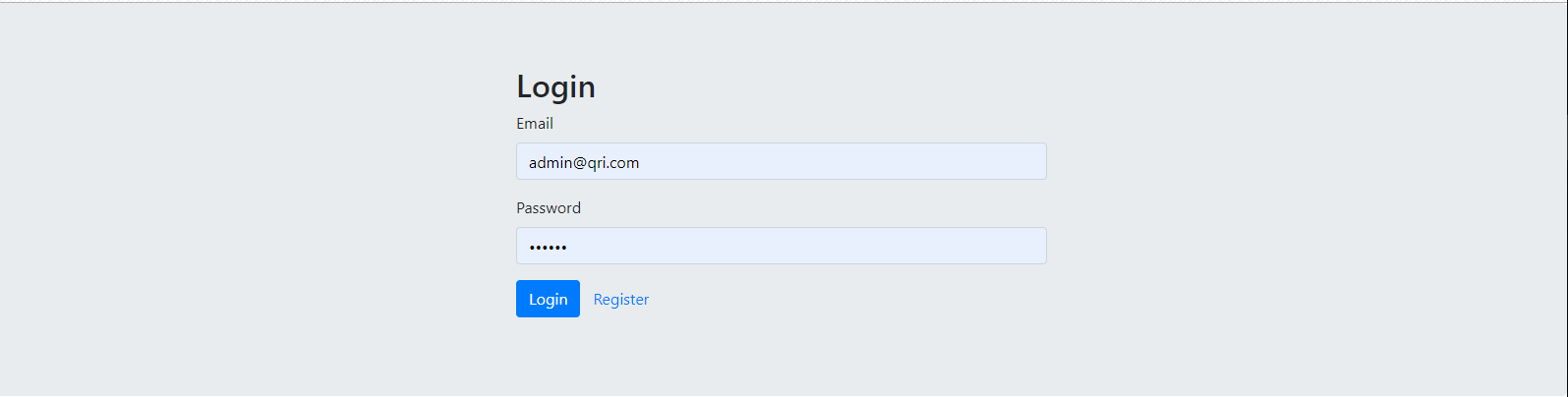
1. Cơ sở dữ liệu phòng học
   * 1. Lưu dữ liệu hình ảnh phòng học



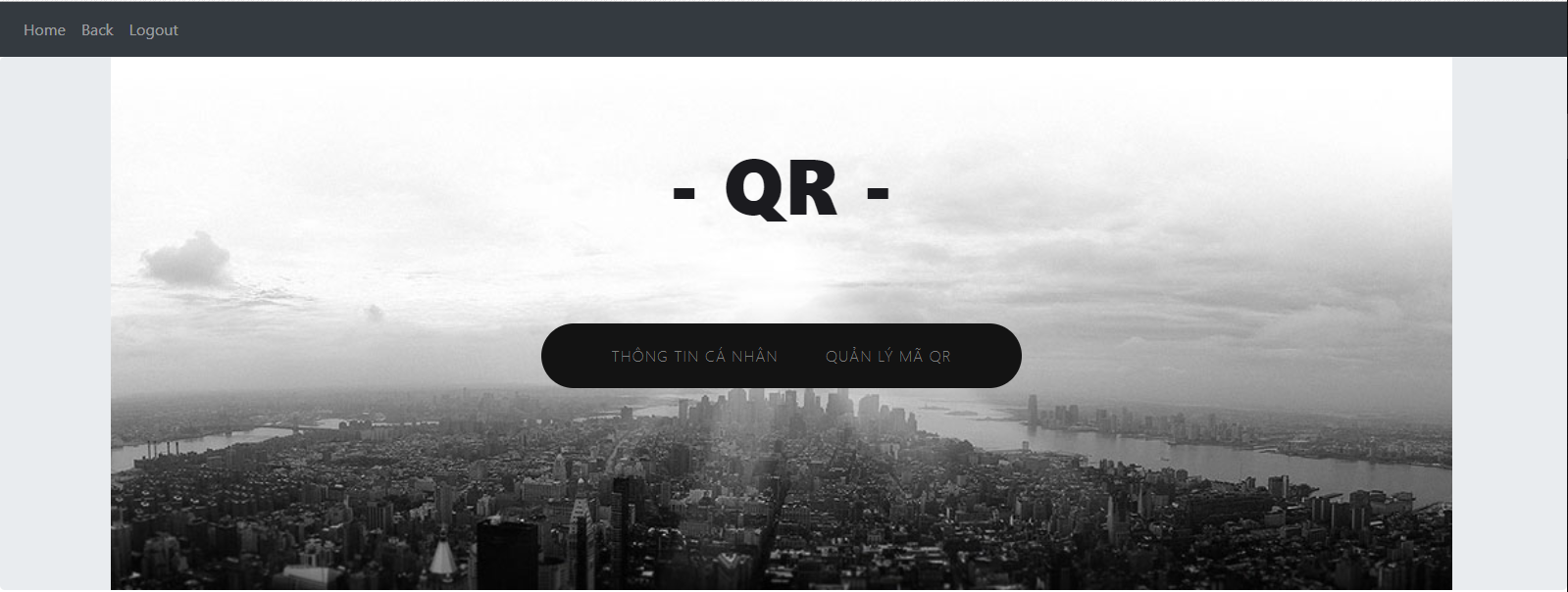
1. Cơ sở dữ liệu hình ảnh phòng học
   * 1. Lưu lịch sử chỉnh sửa



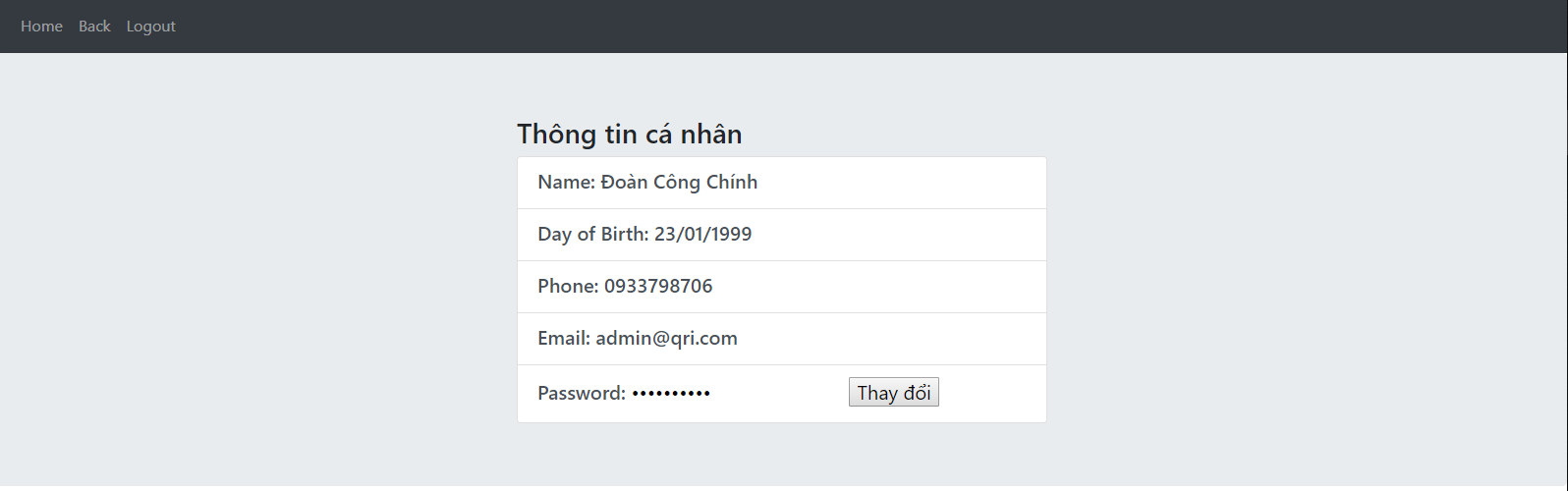
1. Cơ sở dữ liệu lịch sử chỉnh sửa
2. GIAO DIỆN CHƯƠNG TRÌNH
   1. Giao diện trang quản lý
      1. Giao diện đăng nhập



1. Giao diện đăng nhập của hệ thống quản lý mã QR
   * 1. Giao diện Home



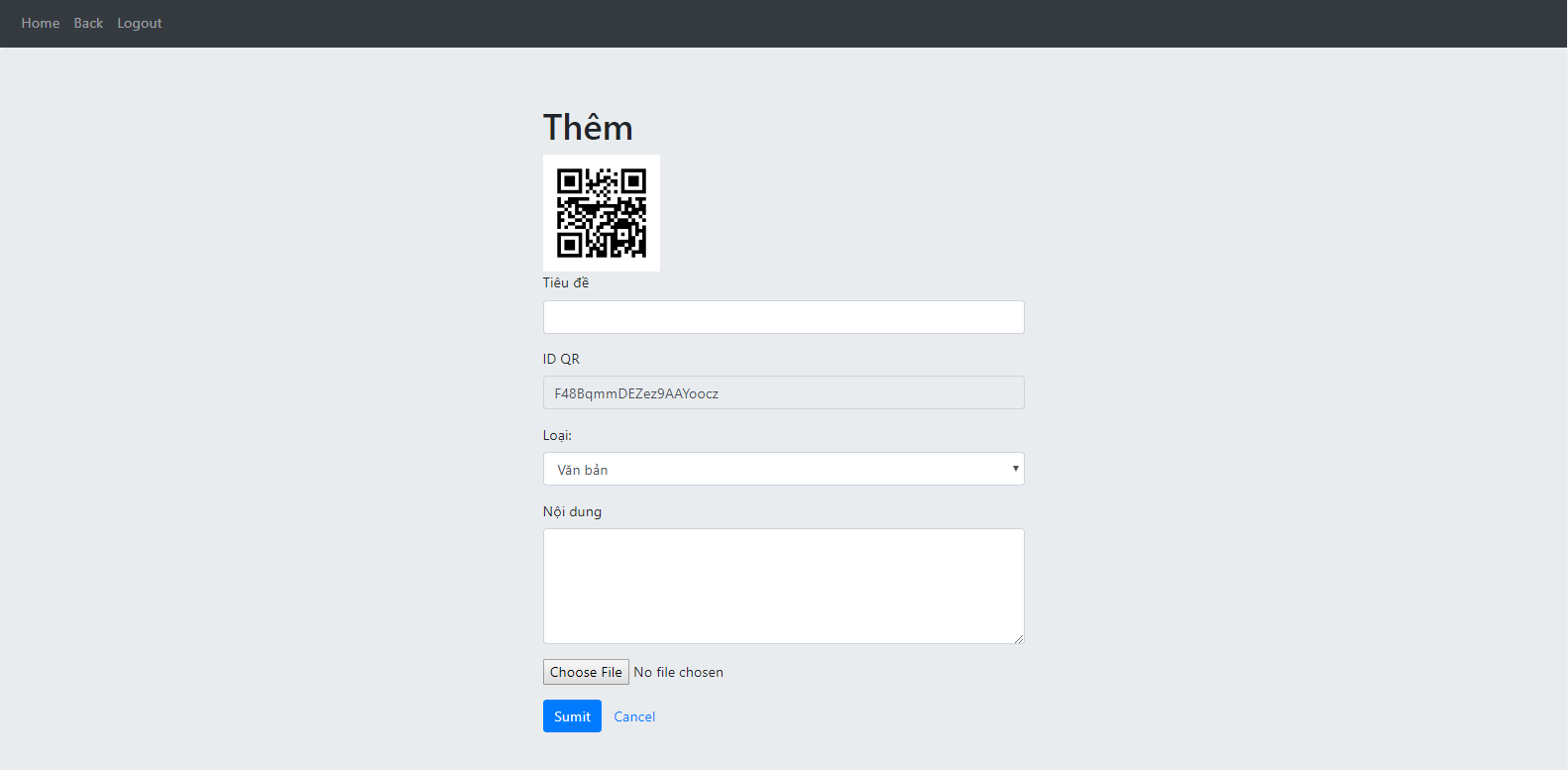
1. Giao diện Home của hệ thống quản lý mã QR
   * 1. Giao diện Thông tin cá nhân



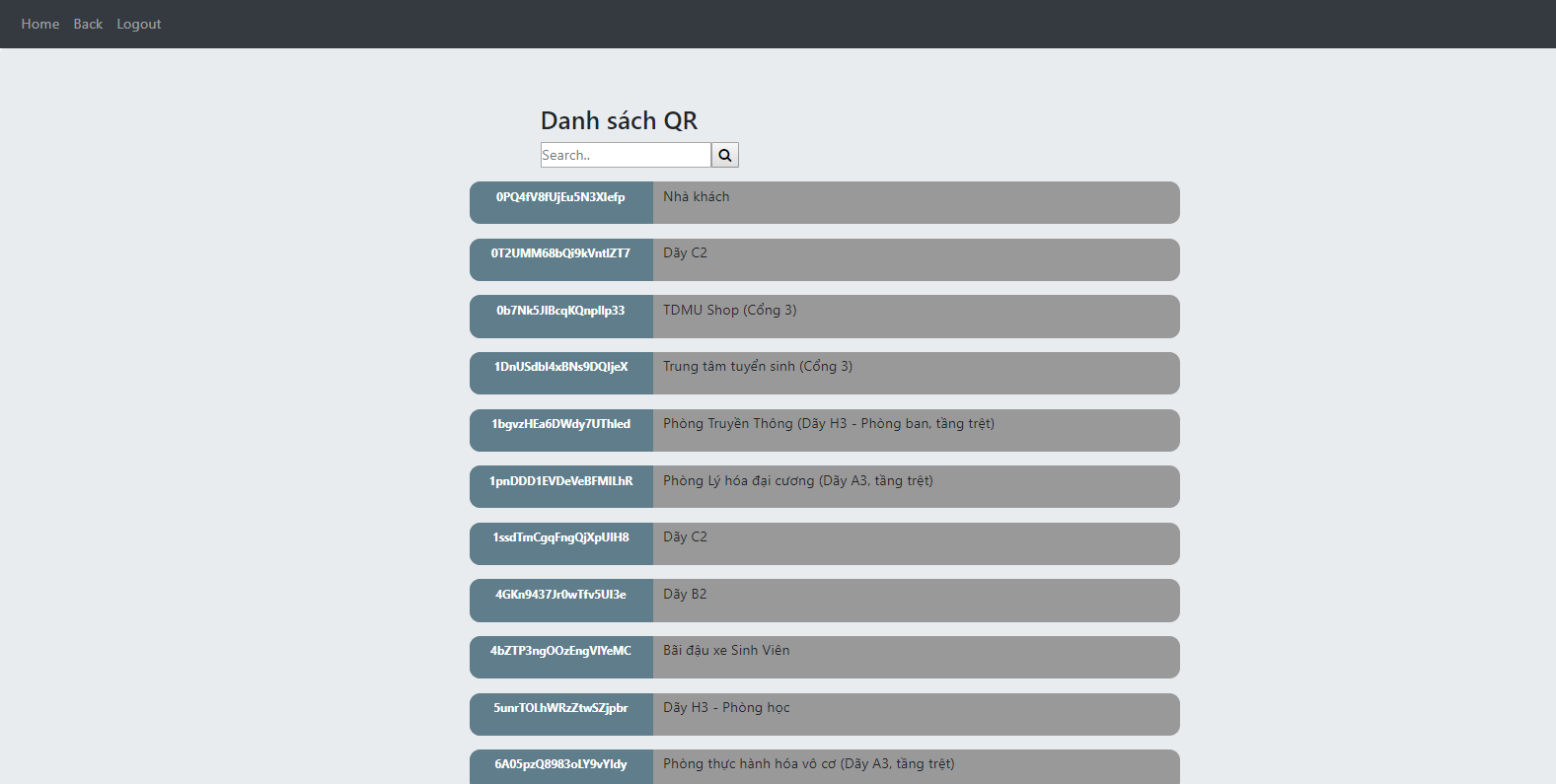
1. Giao diện Thông tin cá nhân của hệ thống quản lý mã QR
   * 1. Giao diện Quản lý mã QR



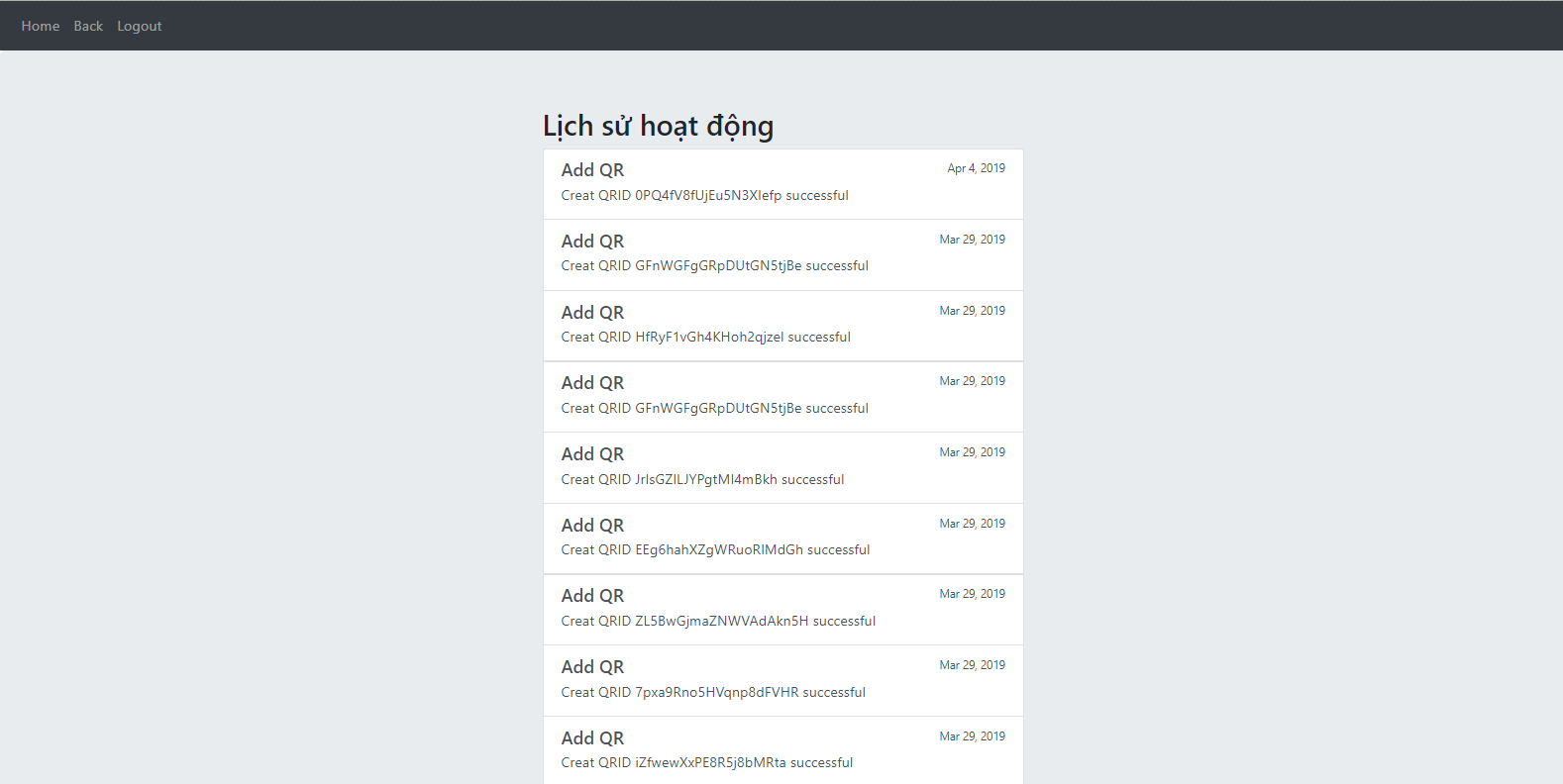
1. Giao diện Quản lý của hệ thống quản lý mã QR
   * 1. Giao diện Thêm mã QR



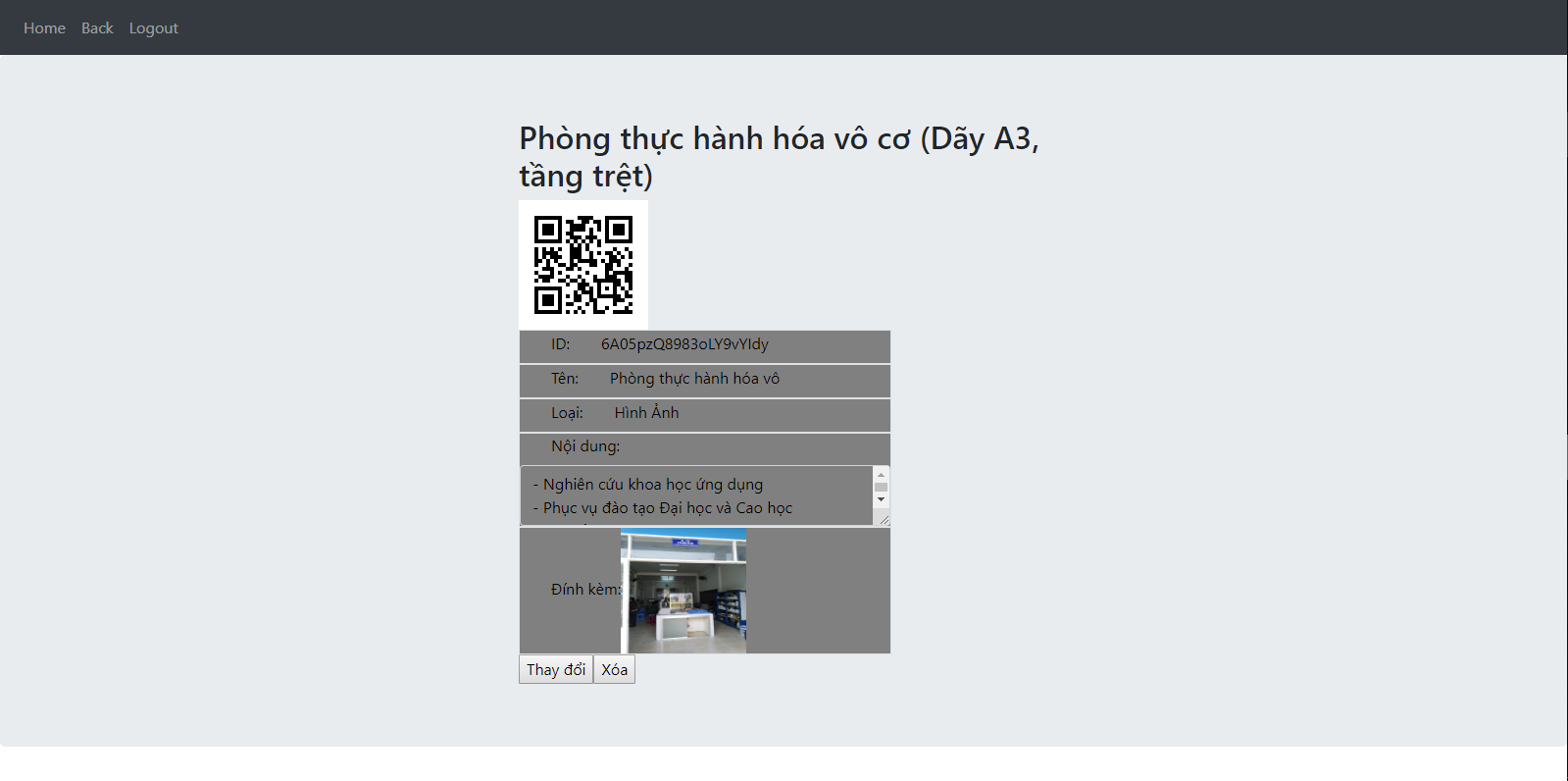
1. Giao diện Thêm mã QR của hệ thống quản lý mã QR
   * 1. Giao diện Danh sách mã QR



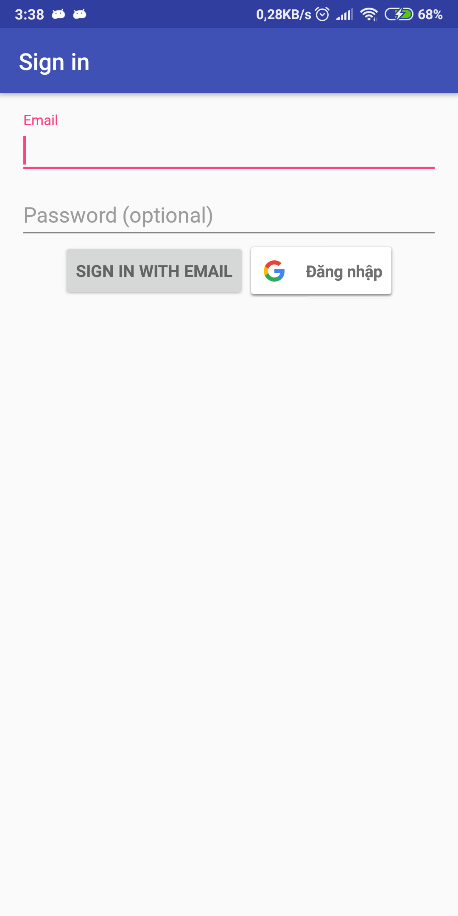
1. Giao diện Danh sách mã QR của hệ thống quản lý mã QR
   * 1. Giao diện Lịch sử



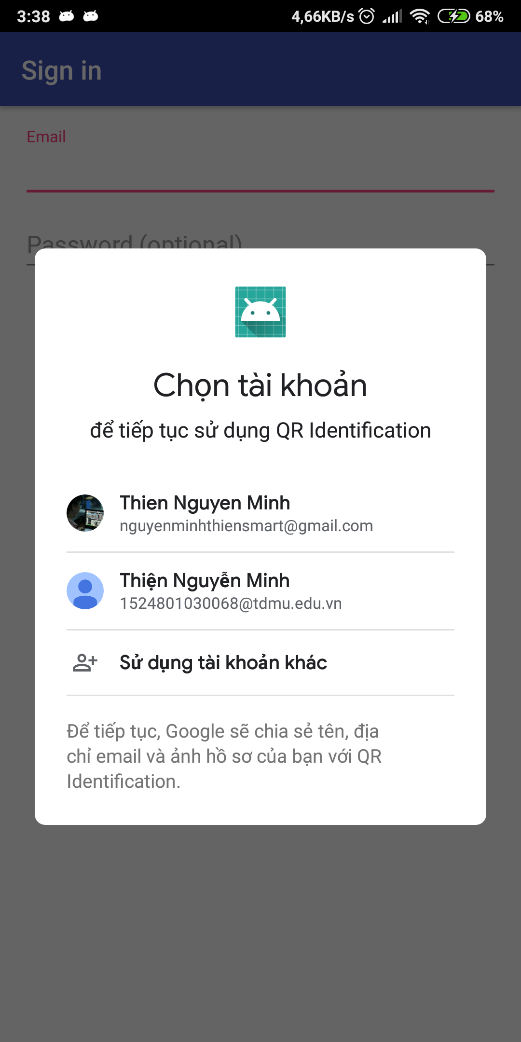
1. Giao diện Lịch sử của hệ thống quản lý mã QR
   * 1. Giao diện Chi tiết mã QR



1. Giao diện xem chi tiết của hệ thống quản lý mã QR
   1. Giao diện ứng dụng trên điện thoại
      1. Giao diện Đăng nhập



1. Giao diện Đăng nhập trên ứng dụng điện thoại



1. Giao diện Đăng nhập trên ứng dụng điện thoại
   * 1. Giao diện Màn hình chính



1. Giao diện Màn hình chính trên ứng dụng điện thoại
   * 1. Giao diện Camera quét mã



1. Giao diện Camera quét mã trên điện thoại
   * 1. Giao diện Hiển thị nội dung kết quả sau khi quét



1. Giao diện Hiển thị nội dung kết quả sau khi quét trên ứng dụng điện thoại
2. KẾT LUẬN
   1. Ưu điểm của đề tài

* Ứng dụng điện thoại gọn nhẹ, dễ dàng đăng nhập qua tài khoản Google.
* Tra cứu thông tin và tìm hiểu sản phẩm một cách đơn giản và nhanh chóng.
* Có khả năng chứa nguồn dữ liệu lớn và truy xuất cực kì nhanh chóng.
* Ứng dụng có khả năng mở rộng để thêm một số tính năng khác.
* Giúp người sử dụng có thể tra cứu nội dung mã QR mọi lĩnh vực mà chỉ cần cài đặt duy nhất một ứng dụng.
  1. Nhược điểm của đề tài
* Giao diện chưa thân thiện với người sử dụng
* Cơ sở dữ liệu sẵn có chưa mở rộng đủ để phát triển đề tài.
* Chưa có khả năng tích hợp các cơ sở dữ liệu bên ngoài
  1. Hướng phát triển của đề tài
* Khắc phục các nhược điểm trên.
* Sử dụng Firebase Analytics trong việc phân tích thị hiếu người dùng.
* Hiển thị một số nội dung liên quan khi người dùng quét mã trên ứng dụng di động.

TÀI LIỆU THAM KHẢO

1. <https://viblo.asia/p/xay-dung-ung-dung-voi-nodejs-amoG81Jbvz8P> truy cập ngày 02/12/2018
2. <https://viblo.asia/p/tong-quan-ve-angular-js-nwmkyMDBkoW> truy cập ngày 05/12/2018
3. <https://vi.wikipedia.org/wiki/Android_(h%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh)> truy cập ngày 12/01/2018
4. <https://firebase.google.com/docs/database/> truy cập ngày 14/03/2018
5. <https://firebase.google.com/docs/auth/> truy cập ngày 14/03/2018
6. <https://firebase.google.com/docs/storage/> truy cập ngày 14/03/2018
7. <https://www.w3schools.com/> truy cập ngày 02/12/2018
8. <https://angular.io/docs> truy cập ngày 05/12/2018