BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CẦN THƠ

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

<TÊN ĐỀ TÀI>

<Mã số đề tài>

Cần Thơ, <Tháng>/<Năm>

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CẦN THƠ

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

<TÊN ĐỀ TÀI>

<Mã số đề tài>

Sinh viên thực hiện: <họ và tên sinh viên> Nam, Nữ:

Dân tộc:

Lớp, khoa: Năm thứ: /Số năm đào tạo:

Ngành học:

(Ghi rõ họ và tên sinh viên chịu trách nhiệm chính thực hiện đề tài)

Người hướng dẫn: <Chức danh khoa học, học vị, họ và tên của người hướng dẫn>

Cần Thơ, <Tháng>/<Năm>

Danh sách những thành viên tham gia nghiên cứu đề tài và đơn vị phối hợp chính

Mục lục

Danh mục bảng biểu

Danh mục những từ viết tắt (xếp theo thứ tự bảng chữ cái)

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CẦN THƠ

THÔNG TIN KẾT QUẢ NGHIÊN CỨU CỦA ĐỀ TÀI

1. Thông tin chung:

**-** Tên đề tài:

- Sinh viên thực hiện:

- Lớp: Khoa: Năm thứ: Số năm đào tạo:

- Người hướng dẫn:

2. Mục tiêu đề tài:

3. Tính mới và sáng tạo:

4. Kết quả nghiên cứu:

5. Đóng góp về mặt kinh tế - xã hội, giáo dục và đào tạo, an ninh, quốc phòng và khả năng áp dụng của đề tài:

6. Công bố khoa học của sinh viên từ kết quả nghiên cứu của đề tài, hoặc nhận xét, đánh giá của cơ sở đã áp dụng các kết quả nghiên cứu *(nếu có)*:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stt | Tên bài báo | Tác giả/  nhóm tác giả | Tên tạp chí | Số tạp chí | Năm  xuất bản |
| 1. |  |  |  |  |  |
| 2. |  |  |  |  |  |

Nếu không có bài báo đăng trên tạp chí thì ghi không và xóa bảng này, xóa ghi ghú này.

|  |  |
| --- | --- |
|  | Ngày tháng năm  Sinh viên chịu trách nhiệm chính  thực hiện đề tài  (ký, họ và tên) |

Nhận xét của người hướng dẫn về những đóng góp khoa học của sinh viên thực hiện đề tài *(phần này do người hướng dẫn ghi):*

|  |  |
| --- | --- |
| Xác nhận của Trường Đại học Cần Thơ  (ký tên và đóng dấu) | Ngày tháng năm  Người hướng dẫn  (ký, họ và tên) |

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CẦN THƠ

THÔNG TIN VỀ SINH VIÊN

CHỊU TRÁCH NHIỆM CHÍNH THỰC HIỆN ĐỀ TÀI

I. SƠ LƯỢC VỀ SINH VIÊN:

Ảnh 4x6

Họ và tên:

Sinh ngày: tháng năm

Nơi sinh:

Lớp: Khóa:

Khoa:

Địa chỉ liên hệ:

Điện thoại: Email:

**II. QUÁ TRÌNH HỌC TẬP** (kê khai thành tích của sinh viên từ năm thứ 1 đến năm đang học):

\* Năm thứ 1:

Ngành học: Khoa:

Kết quả xếp loại học tập:

Sơ lược thành tích:

\* Năm thứ 2:

Ngành học: Khoa:

Kết quả xếp loại học tập:

Sơ lược thành tích:

...

|  |  |
| --- | --- |
| Xác nhận của Trường Đại học Cần Thơ  (ký tên và đóng dấu) | Ngày tháng năm  Sinh viên chịu trách nhiệm chính  thực hiện đề tài  (ký, họ và tên) |

1. Mở đầu
   1. Tổng quan tình hình nghiên cứu thuộc lĩnh vực đề tài ở trong và ngoài nước
   2. Lý do chọn đề tài
   3. Mục tiêu đề tài
   4. Phương pháp nghiên cứu
   5. Đối tượng và phạm vi nghiên cứu
2. Kết quả nghiên cứu

<abstract?>

# Cơ sở lý thuyết

## Flutter

<https://flutter.dev/docs/resources/faq>

### Khái niệm

Flutter là một bộ công cụ phát triển ứng dụng (Software Development Toolkit - SDK) di động đa nền tảng cho Android, iOS, web, Windows, Linux, và Fushchia được phát triển bởi Google. Mục đích chính của Flutter là giúp các nhà phát triển tạo ra những ứng dụng chất lượng cao với giao diện mượt và đẹp mắt trong thời gian ngắn chỉ với một code base, hay nói cách khác, chỉ cần một lần code cho cả hai nền tảng mà không phải tạo thêm một ứng dụng riêng biệt thông qua một ngôn ngữ chuyên dụng khác như Swift, hay C#. Flutter được xây dựng dựa trên C++ và ngôn ngữ lập trình Dart.

Flutter SDK bao gồm:

* Reactive framework với Dart và đa dạng thư viện, API, plugins
* Bộ máy render 2D riêng biệt
* Hệ thống Widgets độc nhất hỗ trợ xây dựng giao diện cho ứng dụng
* Bộ công cụ hỗ trợ phát triển ứng dụng bao gồm debugger, formatter,...

### Ứng dụng thực tế trong đề tài

Ứng dụng quản lý nồng độ cồn tài xế là ứng dụng di động, được phát triển chủ yếu cho hệ điều hành Android. Do Flutter được biên dịch trực tiếp sang native ARM code nhờ những thuộc tính của Dart, việc mở rộng và phát triển thêm cho những nền tảng khác như iOS sẽ trở nên nhanh chóng, dễ dàng, và tiết kiệm hơn.

### Ưu điểm và nhược điểm

#### Ưu điểm của Flutter:

* Tiết kiệm thời gian phát triển ứng dụng:
  + Với các chức năng như hot reload và hot restart
  + Thừa hưởng khả năng biên dịch linh động của Dart với JIT cho quá trình phát triển và AOT cho sản phẩm cuối
  + Phát triển ứng dụng di động cho Andoid và iOS chỉ với một lần code.
* Dễ dàng thiết kế giao diện nhờ khả năng tích hợp layout trong code cùng backend, không cần phải viết giao diện trên một template riêng như JSX hay XML.
* Hiệu suất ứng dụng nhanh và mượt mà trên mọi nền tảng vì không phải phụ thuộc vào render của OEM Widgets ((Thống nhất thành phần giao diện của các nhà cung cấp thiết bị di động).

#### Nhược điểm của Flutter:

* Kích thước ứng dụng lớn do render với widgets của Flutter chứ không phụ thuộc vào OEM widgets.
* Cần phải học ngôn ngữ mới như Dart thay vì những lựa chọn quen thuộc hơn là Java hay JavaScript.

## Dart

### Khái niệm

Dart là ngôn ngữ lập trình đa nền tảng, dùng để xây dựng ứng dụng cho web, máy tính, và thiết bị di động. Dart được phát triển bởi Google dựa trên hệ cú pháp của ngôn ngữ C và tự động được biên phiên dịch (transcompile hay source-to-source compile) sang JavaScript.

Dart sử dụng mẫu hình lập trình hướng đối tượng (Object-oriented hay OOP) và hướng lớp (class-oriented) với cơ chế thu gom rác (garbage collector).

Dart là ngôn ngữ type safe: Kết hợp giữa static type check và runtime check để bảo đảm một giá trị của biến luôn trùng khớp với static type của biến. Tuy nhiên, việc báo cáo kiểu dữ liệu là tùy chọn (optionally typed) nhờ khả năng suy luận dữ liệu của Dart (type inference).

Các thiết lập của Dart được sắp xếp theo từng module khác nhau dưới dạng thư viện (libraries).

(<https://dart.dev/guides/language/specifications/DartLangSpec-v2.2.pdf>)

### Ứng dụng thực tế trong đề tài

Ứng dụng nhân viên nồng độ cồn được phát triển chủ yếu dựa trên Flutter. Dart là ngôn ngữ chính được sử dụng trong Flutter - một framework lập trình di động đa nền tảng hiện đại được phát triển bởi Google.

### Ưu điểm và nhược điểm

#### Ưu điểm của Dart:

* Dart là mã nguồn mở (Open-source), được Google và cộng đồng lập trình viên trên khắp thế giới góp phần phát triển.
* Syntax của Dart tương tự Java, JavaScript và C, cũng như các ngôn ngữ OOP khác, giúp lập trình viên dễ dàng làm quen.
* Dart là ngôn ngữ type safe - nhờ vậy mà giảm thiểu khả năng xảy ra lỗi, giảm thiểu boilerplate code giúp cho code dễ đọc hơn, nên sẽ tối thiểu thời gian phát triển ứng dụng.
* Khả năng biên dịch với cả hai cách Ahead Of Time (AOT) và Just In Time (JIT)

(https://flutter.dev/docs/resources/faq#why-did-flutter-choose-to-use-dart)

* AOT: Trực tiếp biên dịch ngôn ngữ Dart sang ARM Native Code trước khi chạy, giúp ứng dụng khởi động nhanh hơn và có thể ước tính hiệu suất của ứng dụng khi chính thức được triển khai.
* JIT: Giúp chu kỳ phát triển ứng dụng được rút gọn nhờ khả năng biên dịch ở runtime, hay nói cách khác, với JIT, Dart chỉ được biên dịch khi cần sử dụng đến, không cần thiết phải biên dịch lại từ đầu mỗi khi có thay đổi (là nền tảng cho hot reload trong lập trình di động)
* Tốc độ: Dart nhanh gấp khoảng 4 lần Java và gấp khoảng 2 lần so với JavaScript (không có proof, không biết có nên để ko)

#### Nhược điểm của Dart

* Dart là ngôn ngữ mới xuất hiện gần đây, chưa được phổ biến so với những ngôn ngữ lập trình đa nền tảng khác.
* Nguồn tư liệu tham khảo còn hạn chế vì cộng đồng sử dụng Dart còn nhỏ, gây khó khăn trong quá trình tìm hướng giải quyết cho vấn đề.

## Firebase

Firebase là nền tảng phát triển di động thuộc quyền sở hữu của Google. Đây là một mô hình dịch vụ điện toán đám mây, cung cấp các dịch vụ cho người phát triển tích hợp vào các ứng dụng trên web và ứng dụng di động. Firebase hỗ trợ nhiều loại hình dịch vụ khác nhau, trong đề tài này, nhóm phát triển đã sử dụng 3 dịch vụ là cơ sở dữ liệu (CSDL) thời gian thực (Firebase Real-time Database), dịch vụ xác thực tài khoản (Firebase Authentication) cùng với dịch vụ chức năng đám mây (Google Cloud Function)

### Firebase Real-time Database

Firebase Real-time Database (Firebase RD) là một CSDL được lưu trữ trên đám mây và là một CSDL dạng NoSQL. Với CSDL này, dữ liệu sẽ được lưu trữ dưới dạng JSON và được đồng bộ hóa trong thời gian thực cho mọi thiết bị sử dụng. Firebase cũng sẽ đồng bộ hóa các thay đổi cục bộ với các cập nhật mới đã xảy ra trong thời gian các thiết bị khách ở chế độ ngoại tuyến một khi các thiết bị này có kết nối lại.

#### Cấu trúc dữ liệu trong Firebase RD: cây JSON (JSON tree)

Tất cả dữ liệu trong Firebase RD được lưu trữ dưới dạng đối tượng JSON được lưu trữ trên hệ thống đám mây. Khác với CSDL SQL, Firebase RD sẽ không có các khái niệm bảng, hàng, cột. Khi dữ liệu được thêm vào cây JSON, nó sẽ được lưu dưới dạng một nút (node) trong cấu trúc JSON gồm 2 thành phần: khóa và dữ liệu (key – data). Khóa có thể được đặt bởi người phát triển hoặc được tạo tự động.

Ví dụ về cấu trúc dữ liệu trong Firebase RD:

{  
  "users": {  
    "alovelace": {  
      "name": "Ada Lovelace",  
      "contacts": { "ghopper": true },  
    },  
    "ghopper": { ... },  
    "eclarke": { ... }  
  }  
}

<https://firebase.google.com/docs/database/ios/structure-data>

Với cơ chế hoạt động như trên, Firebase RD phù hợp với các ứng dụng như live streaming, nhắn tin, định vị GPS, … và đem lại các lợi thế như sau:

* Hỗ trợ lưu trữ thay đổi khi người dùng sử dụng ở chế độ offline, và thực hiện đồng bộ ngay khi người dùng có kết nối Internet.
* Tích hợp với Firebase Auth để xây dựng cơ chế bảo mật đơn giản mà vẫn hiệu quả
* Không tốn chi phí để xây dựng và bảo trì backend server
* Hỗ trợ đồng bộ thời gian thực và hiển thị dữ liệu nhanh chóng trên các thiết bị client
* Sử dụng cấu trúc dữ liệu dạng NoSQL, đem lại tốc độ nhanh chóng

### Firebase Authentication

Firebase Auth là dịch vụ hỗ trợ xác thực 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 trên từng thiết bị của các người dùng khác nhau. Để làm được điều đó, Firebase Authentication hỗ trợ xác thực bằng mật khẩu, số điện thoại hoặc thông qua sử dụng các tài khoản xã hội phổ biến như Google, Facebook, Twitter, … và được đảm bảo về bảo mật bởi đội ngũ phát triển  từ Google.

Trong phạm vi đề tài, tài khoản của người dùng sẽ ở dạng email được cấp bởi chủ sở hữu dự án (khác với việc người dùng có thể dùng tài khoản email cá nhân để đăng ký).

### Google Cloud function

Google Cloud Function là giải pháp tính toán không cần máy chủ của Google để tạo các ứng dụng hướng sự kiện. Nó là sản phẩm kết hợp giữa đội ngũ phát triển của Google Cloud Platform và Firebase.

Đối với người sử dụng Firebase, Cloud Function cho phép mở rộng các dịch vụ mà Firebase cung cấp. Trong phạm vi đề tài, Cloud Function được sử dụng để hỗ trợ việc gửi thông báo đến ứng dụng di động của người dùng cũng như cập nhật CSDL theo thời gian thực khi có các sự kiện đặc biệt diễn ra

## NoSQL

## Real-time

## IoT

## BAC

### Khái niệm https://en.wikipedia.org/wiki/Blood\_alcohol\_content

Blood Alcohol Concentration (BAC) - Nồng độ cồn trong máu là khái niệm sử dụng phổ biến nhất như một thước đo ngộ độc rượu cho các mục đích pháp lý hoặc y học.

Nồng độ cồn trong máu thường được biểu thị bằng phần trăm ethanol trong máu tính theo đơn vị khối lượng rượu trên một thể tích máu hoặc khối lượng rượu trên một khối lượng máu, tùy thuộc vào quốc gia.

Ví dụ, BAC là 0.10 (0.10%) có nghĩa là có 0,10g (100mg) rượu cho mỗi 100 ml máu

### Tác động của cồn theo chỉ số BAC

Table 1. Tác động của cồn theo chỉ số BAC https://alcohol.stanford.edu/alcohol-drug-info/buzz-buzz/what-bac

|  |  |
| --- | --- |
| Chỉ số BAC | Tác động lên cơ thể và hệ thần kinh |
| 0.01 - 0.03 | Không có tác động rõ ràng, có thể có một chút hưng phấn. |
| 0.04 - 0.06 | Có cảm giác thư giãn và ấm áp. Khả năng tư duy và trí nhớ bị ảnh hưởng nhẹ. |
| 0.07 - 0.09 | Khả năng thăng bằng, ngôn ngữ, tầm nhìn và điều khiển hành động bắt đầu bị ảnh hưởng. |
| 0.10 - 0.12 | Sự phối hợp vận động của các chi bị suy giảm đáng kể và gần như mất khả năng phán đoán. Khả năng ngôn ngữ bị ảnh hưởng, phát ngôn không rõ ràng |
| 0.13 - 0.15 | Khả năng hành động tổng thể bị suy giảm. Tầm nhìn mờ và mất khả năng thăng bằng lớn. Có thể xuất hiện tình trạng rối loạn ý thức kèm theo tâm trạng lo lắng, bồn chồn |
| 0.16 - 0.20 | Rối loạn ý thức trở nên rõ ràng hơn. Có thể có dấu hiệu buồn nôn. Xuất hiện các dấu hiệu “say xỉn” |
| 0.25 - 0.30 | Nhiễm độc cồn. Đi đứng không vững, cần người hỗ trợ. Ý thức rối loạn, buồn nôn và nôn mửa |
| 0.35 - 0.40 | Mất ý thức, có khả năng hôn mê. |
| 0.40 trở lên | Hôn mê, có khả năng gây tử vong do suy hô hấp |

### Nồng độ cồn liên quan đến việc lái xe ô tô theo quy định của pháp luật

Table 2. Bảng tổng hợp mức phạt vi phạm nồng độ cồn đối với xe ô tô https://nganhangphapluat.thukyluat.vn/tu-van-phap-luat/vi-pham-hanh-chinh/muc-phat-nong-do-con-khi-lai-xe-nam-2019-280843

|  |  |  |  |
| --- | --- | --- | --- |
| Bảng tổng hợp mức phạt vi phạm nồng độ cồn đối với xe ô tô | | | |
| Nồng độ | Cơ sở pháp lý | Mức phạt tiền | Hình thức phạt bổ sung |
| <50mg/100ml máu | K6, Đ5, Nghị định 46/2016/NĐ-CP(\*) | 2 triệu - 3 triệu VNĐ | Tạm giữ phương tiện 7 ngày  Tước giấy phép lái xe 1 đến 3 tháng |
| <0.25mg/1L khí thở |
| <0.05 BAC |
| 50-80mg/100ml máu | K8, Đ5,  Nghị định 46/2016/NĐ-CP | 7 triệu - 8 triệu VNĐ | Tạm giữ phương tiện 7 ngày  Tước giấy phép lái xe 3 đến 5 tháng |
| 0.25-0.4mg/1L khí thở |
| 0.05-0.08 BAC |
| >80mg/100ml máu | K9, Đ5,  Nghị định 46/2016/NĐ-CP | 16 triệu - 18 triệu VNĐ | Tạm giữ phương tiện 7 ngày  Tước giấy phép lái xe 4 đến 6 tháng |
| >0.4mg/1L khí thở |
| >0.08 BAC |

(\*): Luật không ghi rõ mốc tối thiểu xác định “có nồng độ cồn”: “Điều khiển xe trên đường mà trong máu hoặc hơi thở có nồng độ cồn nhưng chưa vượt quá 50 miligam/100 mililít máu hoặc chưa vượt quá 0,25 miligam/1 lít khí thở.”

### Tính giá trị BAC đọc được từ MQ3

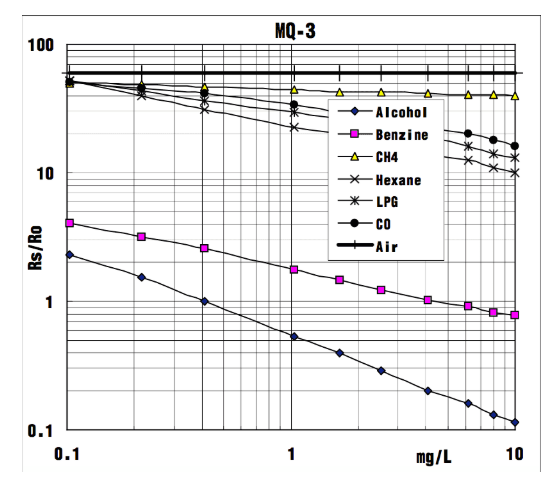


Figure 1. Đồ thị giá trị tỉ lệ Rs/R0  
Nguồn: https://www.sparkfun.com/datasheets/Sensors/MQ-3.pdf

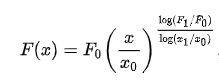
Đồ thị giá trị tỉ lệ Rs/R0 với

* Rs là giá trị điện trở của cảm biến khi đo nồng độ cồn trong không khí
* R0 là hằng số mang giá trị điện trở của cảm biến khi nồng độ cồn trong không khí là có 0.4mg/L

Dựa vào đồ thị trên, ta nhận thấy giá trị Rs/R0là hằng số có giá trị 60 trong điều kiện không khí sạch, và theo định nghĩa, chỉ có giá trị Rs thay đổi trong khi R0 là hằng số. Do đó, nếu ta đo được giá trị Rs trong không khí sạch, ta có thể xác định giá trị R0 và tỉ lệ Rs/R0 trong điều kiện đo số liệu thông thường.

Khi đã có giá trị Rs trong điều kiện không khí sạch (\*footnote: chi tiết cách đo ở https://www.teachmemicro.com/mq-3-alcohol-sensor/) , ta cần xác định mối quan hệ giữa giá trị BAC và tỉ lệ Rs/R0.

Ta có phương trình tổng quát của một biểu đồ log-log là



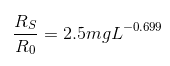
với F1, x1 và F0, x0 là tọa độ của 2 điểm trên đồ thị. Nếu ta xem xét đồ thị giá trị Rs/R0 trên đường biểu diễn *Alcohol* trong biểu đồ(ref len biểu đồ) phía trên, ta có thể định vị được 2 điểm có tọa độ xấp xỉ như sau:

* F0 = 1, x0 = 0.4
* F1 = 0.2, x1 = 4

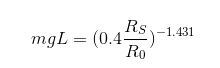
Thay tọa độ vào phương trình tổng quát ta có



Như vậy,



Vậy ta có thể dùng biểu thức sau để tìm ra giá trị nồng độ cồn đo được ở đơn vị mg/L



Để phù hợp với đơn vị đo thông thường, ta có thể chuyển đổi về đơn vị g/100ml bằng cách nhân kết quả thu được với 0.0001

https://www.teachmemicro.com/mq-3-alcohol-sensor/

# Nội dung nghiên cứu

## Phát triển ứng dụng di động với Flutter

## Flutter và Firebase API

### Firebase RD với Flutter

<https://medium.com/flutterpub/realtime-database-in-flutter-bef0f29e3378>

<https://firebase.google.com/docs/reference/android/com/google/firebase/database/FirebaseDatabase>

<https://firebase.google.com/docs/reference/android/com/google/firebase/database/DatabaseReference.html>

#### Khởi tạo và tham chiếu đến CSDL

Đối với một ứng dụng di động được phát triển bằng framework Flutter, CSDL Firebase RD sẽ được truy cập vào dưới dạng một thể hiện (instance) của lớp FirebaseDatabase. Để lấy được dữ liệu trong CSDL, ta cần tham chiếu (reference) đến dữ liệu trong CSDL thông qua hàm reference() trên instance đã có, hàm sẽ trả về tham chiếu đến cây CSDL (database) để ta có thể thực hiện các hoạt động đọc và ghi dữ liệu.

FirebaseDatabase.instance.reference()

Nếu muốn thao tác trên các nhánh con (dưới dạng URL) của cây CSDL, ta có thể sử dụng phương thức child() với tham số là tên nhánh muốn truy cập:

FirebaseDatabase.instance.reference().child('driver')

Nếu nhánh con được yêu cầu chưa tồn tại, nhánh đó sẽ được khởi tạo nếu có yêu cầu ghi mới CSDL

#### Thao tác ghi dữ liệu

Thao tác ghi dữ liệu được thực hiện thông qua phương thức set() với tham số đầu vào là dữ liệu theo cấu trúc JSON

FirebaseDatabase.instance.reference().child('driver').child(newID).child('basicInfo')

.set({

'name' : \_nameController.text,

'idCard' : \_idCardController.text,

'address' : \_addressController.text,

'gender' : genderRadioGroup == 0 ? 'M' : 'F',

'dob' : dob,

'email' : newID.toLowerCase() + '@driver.potatoes.com',

});

#### Thao tác đọc dữ liệu

Firebase RD cho phép đọc dữ liệu dưới dạng “stream” - luồng dữ liệu theo thời gian thực. Điều đó có nghĩa là mỗi khi có thay đổi được thực hiện trên trung tâm CSDL, các client có kết nối Internet đều sẽ được cập nhật thông tin tự động. Thư viện cho Firebase của Flutter cho phép sử dụng stream onValue để thực hiện thao tác đọc real-time này:

FirebaseDatabase.instance.reference().child('driver').onValue

Để hiển thị dữ liệu cho người dùng, Flutter hỗ trợ widget StreamBuilder (widget làm việc với các snapshot nhận được từ stream).

StreamBuilder(

stream: FirebaseDatabase.instance.reference().child('driver')

.orderByChild('isDeleted').equalTo(false)

.onValue,

builder: (BuildContext context, snapshots) {

//return widget

},

)

Các đối tượng đồ họa sẽ được xây dựng ở builder

Ngoài onValue được dùng để theo dõi tất cả các thay đổi trên CSDL, ta cũng có thể sử dụng các cấu hình stream như sau để lắng nghe trên một số sự kiện nhất định

|  |  |
| --- | --- |
| onChildAdded | Kích hoạt khi có nút con mới được thêm vào |
| onChildChange | Kích hoạt khi dữ liệu có sự thay đổi dữ liệu trên nút con |
| onChildRemoved | kích hoạt khi có nút con bị xóa |

https://firebase.google.com/docs/reference/android/com/google/firebase/database/ChildEventListener

Hoặc ta có thể dùng hàm once() để nhận dữ liệu 1 lần và không quan tâm đến các thay đổi (nếu có) sau đó

FirebaseDatabase.instance.reference().child('driver').once()

#### Thao tác cập nhật dữ liệu

Thao tác ghi dữ liệu được thực hiện thông qua phương thức update() với tham số đầu vào là dữ liệu theo cấu trúc JSON

FirebaseDatabase.instance.reference().child('driver').child(dID).child('basicInfo')

.update({

'name' : \_nameController.text,

'idCard' : \_idCardController.text,

'address' : \_addressController.text,

'gender' : genderRadioGroup == 0 ? 'M' : 'F',

'dob' : dob,

});

#### Thao tác xóa dữ liệu

Thao tác ghi dữ liệu được thực hiện thông qua ham2 remove()

//remove() is equivalent to calling set(null)

FirebaseDatabase.instance.reference().child('trips').child(tID).remove();

Hoặc có thể gọi hàm set() hoặc update() như ở thao tác ghi và cập nhật dữ liệu, với giá trị tham số đầu vào là giá trị null

FirebaseDatabase.instance.reference().child('trips').child(tID).set(null);

Cách dùng với update có thể được sử dụng khi muốn xóa một (hoặc một số) dữ liệu trong một nút con mà không xóa toàn bộ nút

FirebaseDatabase.instance.reference().child('trips').child(tID).update({'dID': null});

### Firebase Authentication với Flutter

#### Khởi tạo

Tương tự như ở Firebase RD, ta cần khởi tạo một instance với Firebase Auth để thực hiện các thao tác trên đối tượng này

FirebaseAuth.instance

#### Thao tác đăng ký và đăng nhập

Khi đã có instance của Firebase Auth, ta có thể thực hiện tạo tài khoản và đăng nhập thông qua các phương thức sau

|  |  |
| --- | --- |
| createUserWithEmailAndPassword (String email, String password) | Tạo tài khoản với thông tin email và password được nhập vào. Nếu tài khoản được tạo thành công, người dùng sẽ được cho phép đăng nhập vào hệ thống ngay sau khi tạo |
| signInWithEmailAndPassword (String email, String password) | Đăng nhập với thông tin email và password được nhập vào.  Trả về kết quả ngoại lệ FirebaseAuthInvalidUserException khi không tìm thấy tài khoản với email tương ứng hoặc FirebaseAuthInvalidCredentialsException khi mật khẩu nhập vào không chính xác |
| signInAnonymously() | Đăng nhập ẩn danh |

#### Thao tác trên thông tin tài khoản

Để thao tác trên thông tin tài khoản, ta sử dụng hàm getCurrentUser(). Hàm sẽ trả về một đối tượng FirebaseUser với thông tin người dùng đang đăng nhập (hoặc giá trị null nếu không có tài khoản nào đang đăng nhập). Đối tượng FirebaseUser cũng cung cấp các phương thức để truy cập hoặc thay đổi thông tin người dùng và nhân viên trạng thái xác thực(authentication) của người dùng. Trong đề tài này, phương thức sendPasswordResetEmail() được sử dụng để cung cấp cho người dùng dịch vụ thay đổi mật khẩu.

|  |  |
| --- | --- |
| sendPasswordResetEmail (String email) | Phương thức này trả về kết quả ngoại lệ FirebaseAuthInvalidUserException khi không tìm thấy tài khoản với email tương ứng |

### Firebase Cloud Messaging với Flutter

Trong quá trình phát triển đề tài, nhóm không sử dụng trực tiếp dịch vụ FCM mà sử dụng gián tiếp thông qua Cloud Function platform (sẽ được đề cập trong phần kế tiếp). Các thông tin cơ sở cần nắm về FCM để có thể áp dụng được khi cài đặt cấu hình dịch vụ ở Cloud Function platform được trình bày ở mục này.

<https://fireship.io/lessons/flutter-push-notifications-fcm-guide/>

<https://firebase.google.com/docs/cloud-messaging/android/client>

#### Các loại tin(messaging type)

<https://firebase.google.com/docs/cloud-messaging/concept-options>

Khi sử dụng FCM, có 2 loại tin có thể được gửi đến thiết bị client:

* Tin thông báo (notification messages): chứa các thuộc tính thông tin cố định đã được định nghĩa sẵn (gồm title và body). FCM hỗ trợ hiển thị các thông tin của loại tin này đến khay thông báo (notification tray) của các thiết bị client

var msg = {

notification: {

title: 'Vượt mức chỉ số cồn',

body: 'Tài xế ' + dID + ' có dấu hiệu vượt mức nồng độ cồn',

}

}

* Tin dữ liệu (data messages): chứa các thuộc tính thông tin do người phát triển tự định nghĩa, đồng thời người phát triển cần tự xây dựng các phương pháp xử lý với loại tin dữ liệu này

var msg = {

data : {

'click\_action': 'FLUTTER\_NOTIFICATION\_CLICK',

'dID' : dID.toString(),

'tripID' : curTripID.toString()

}

}

Cả 2 loại tin có thể được sử dụng cùng nhau dưới dạng gửi đi một tin thông báo có kèm tin dữ liệu

var msg = {

notification: {

title: 'Vượt mức chỉ số cồn',

body: 'Tài xế ' + dID + ' có dấu hiệu vượt mức nồng độ cồn',

},

data : {

'click\_action': 'FLUTTER\_NOTIFICATION\_CLICK',

'dID' : dID.toString(),

'tripID' : curTripID.toString()

}

}

#### Nhận tin trên ứng dụng di động

Đối với việc nhận thông báo trên ứng dụng, ta cần quan tâm đến 3 loại trạng thái

* Foreground: là trạng thái ứng dụng đang được sử dụng và đang được hiển thị trên màn hình
* Background: là trạng thái ứng dụng đang chạy ngầm (ứng dụng không được hiển thị trên màn hình nhưng vẫn đang trong trạng thái sử dụng)
* Terminated: ứng dụng đã bị tắt hoàn toàn

Khi tin được gửi tới ứng dụng trong trạng thái foreground, chúng ta có thể xử lý trực tiếp thông qua một hộp thoại (dialog) để hiển thị nội dung thông báo.

Đối với trường hợp ứng dụng đang chạy ngầm hoặc đã tắt, tin sẽ được gửi đến khay hệ thống (system tray) cùng với tên, biểu tượng của ứng dụng và nội dung của tin thông báo. Khi người dùng chạm (tap) vào thông báo, cách thao tác lệnh trong các callback onResume và onLaunch sẽ được thực thi tùy vào trạng thái của ứng dụng (hoặc trong trường hợp mặc định, ứng dụng sẽ được mở lên và hiển thị trang chính), khi đó, các thao tác xử lý trên tin dữ liệu có thể được thực thi

Table 3. Các trường hợp nhận tin

|  |  |  |  |
| --- | --- | --- | --- |
| Trạng thái  Loại tin | Foreground | Background | Terminated |
| Tin thông báo | Thực thi các thao tác liên quan  đã cài đặt trong callback *onMessage.* | Thông báo được gửi đến khay hệ thống của thiết bị client.  Khi người dùng tap vào thông báo, ứng dụng sẽ thực thi các thao tác liên quan  đã cài đặt trong callback *onResume* nếu thuộc tính *'click\_action': 'FLUTTER\_NOTIFICATION \_CLICK'* được cài đặt*.* | Thông báo được gửi đến khay hệ thống của thiết bị client.  Khi người dùng tap vào thông báo, ứng dụng sẽ thực thi các thao tác liên quan  đã cài đặt trong callback *onLaunch* nếu thuộc tính *'click\_action': 'FLUTTER\_NOTIFICATION \_CLICK'* được cài đặt*.* |
| Tin dữ liệu | Thực thi các thao tác liên quan đã cài đặt trong callback *onMessage.* | Khi người dùng tap vào thông báo, ứng dụng sẽ thực thi các thao tác liên quan đã cài đặt trong callback *onResume* nếu thuộc tính *'click\_action': 'FLUTTER\_NOTIFICATION \_CLICK'* được cài đặt | Khi người dùng tap vào thông báo, ứng dụng sẽ thực thi các thao tác liên quan đã cài đặt trong callback *onLaunch* nếu thuộc tính *'click\_action': 'FLUTTER\_NOTIFICATION \_CLICK'* được cài đặt |

#### Gửi và nhận tin theo cơ chế chủ đề

Dựa trên mô hình publish/ subscribe, FCM hỗ trợ gửi cơ chế gửi thông báo cho các thiết bị client đã đăng ký subscribe trên một chủ đề (topic) nhất định. Nói các khác, khi sử dụng cơ chế topic, hệ thống sẽ không gửi thông báo dư thừa đến các thiết bị không đăng ký topic đó.

\_fcm.subscribeToTopic('alcoholTracking');

Hoặc hủy đăng ký topic

\_fcm.unSubscribeFromTopic('alcoholTracking');

Trong đó, \_fcm là một instance của đối tượng FirebaseMessaging trong Flutter.

### Cloud function

<https://firebase.google.com/docs/functions/database-events>

#### Thao tác với Firebase RD

* Thao tác trên sự kiện: Cloud function hỗ trợ lắng nghe và thực thi các thao tác xử lý trên CSDL Firebase RD với 2 cấp độ cụ thể
* Một sự kiện cụ thể: tạo mới, thay đổi hoặc xóa
* Bất kì sự kiện thay đổi dữ liệu nào

|  |  |
| --- | --- |
| onWrite() | kích hoạt khi có sự thay đổi dữ liệu bất kì (tạo mới, cập nhật hoặc xóa) |
| onCreate() | chỉ kích hoạt khi có dữ liệu được tạo mới |
| onUpdate() | chỉ kích hoạt khi có dữ liệu được cập nhật |
| onDelete() | chỉ kích hoạt khi có dữ liệu bị xóa |

Đối với sự kiện onWrite và onUpdate, ta có thể làm việc với cả 2 ảnh chụp (snapshot) của dữ liệu trước và sau khi thay đổi.

Đối với sự kiện onWrite và onUpdate, ta có thể làm việc với cả 2 ảnh chụp (snapshot) của dữ liệu trước và sau khi thay đổi.

functions.database.ref('driver/{dID}/notiInfo/alcoholVal').onUpdate((change, context) => {

const preAlcoVal = change.before.val();

const newAlcoVal = change.after.val();

//other processing…

}

Đối với sự kiện onWrite và onUpdate, ta sẽ làm việc trên 1 snapshot duy nhất của dữ liệu được tạo mới hoặc đã bị xóa.

* Thao tác trên đường dẫn (path)

Để cài đặt cơ chế lắng nghe và kích hoạt sự kiện ở một nhánh dữ liệu cụ thể, ta sử dụng hàm ref() với tham số vào là đường dẫn đến nhánh dữ liệu đó.

functions.database().ref("/driver/TX0003")

Các thao tác dữ liệu xảy ra trên các nhánh con của đường dẫn trên cũng sẽ kích hoạt các sự kiện tương ứng mà ta cài đặt. Ví dụ, khi sử dụng đường dẫn ref("/driver/TX0003"), các sự kiện trên "/driver/TX0003/basicInfo" cũng sẽ được kích hoạt tương ứng và trả về snapshot của dữ liệu ở đường dẫn gốc "/driver/TX0003".

Ta cũng có thể sử dụng ký tự đại diện khi làm việc với đường dẫn thông qua việc sử dụng ngoặc nhọn {}. Cụ thể, nếu ta sử dụng ref("/driver/{dID}"), hệ thống sẽ lắng nghe sự kiện trên bất kì nhánh con nào của “/driver” và chỉ trả về snapshot của dữ liệu ở nhánh con kích hoạt sự kiện. Ví dụ, nếu có thay đổi dữ liệu trên “/driver/TX0001”, hệ thống sẽ trả về snapshot của dữ liệu ở đường dẫn con "/driver/TX0001". Đồng thời, ta có thể sử dụng ký tự đại diện dID như một tham số cho các thao tác thực thi

functions.database.ref('driver/{dID}/notiInfo/alcoholVal').onUpdate((change, context) => {

const dID = context.params.dID;

//other processing…

}

* Thao tác trên dữ liệu

Để thao tác trực tiếp trên dữ liệu, ta có thể sử dụng bộ công cụ admin SDK hỗ trợ bởi Cloud Function cùng các hàm sau:

|  |  |
| --- | --- |
| on | lắng nghe thay đổi của dữ liệu theo thời gian |
| once | truy xuất dữ liệu 1 lần |
| set | tạo mới dữ liệu  \*nếu sử dụng giá trị *null* cho dữ liệu được tạo mới thì thao tác này tương đương với việc xóa dữ liệu (remove) |
| update | cập nhật dữ liệu |

Ví dụ:

admin.database().ref("/bnotification/" + notiID).set({

body: "Tài xế " + dID + " có dấu hiệu vượt mức nồng độ cồn",

dID : dID,

isSolved: false,

timeCreated: lastNotiTime,

tripID: curTripID

})

### Firebase Cloud Messaging

Admin SDK của Cloud function cũng hỗ trợ làm việc với FCM nhằm mục đích tự động gửi thông báo đến cho các thiết bị client khi lắng nghe được các event đặc biệt đã được cài đặt sẵn. Trong đề tài, hàm sendToTopic được sử dụng với tham số đầu tiên là topic và tham số thứ 2 là payload của nội dung tin muốn gửi đi.

admin.messaging().sendToTopic('alcoholTracking', msg)

Trong đó, msg là payload tin đã được định nghĩa trước như trong nội dung đã trình bày với cấu trúc tin của FCM

## Thiết kế hệ thống

đổi hìnhhhhhhh

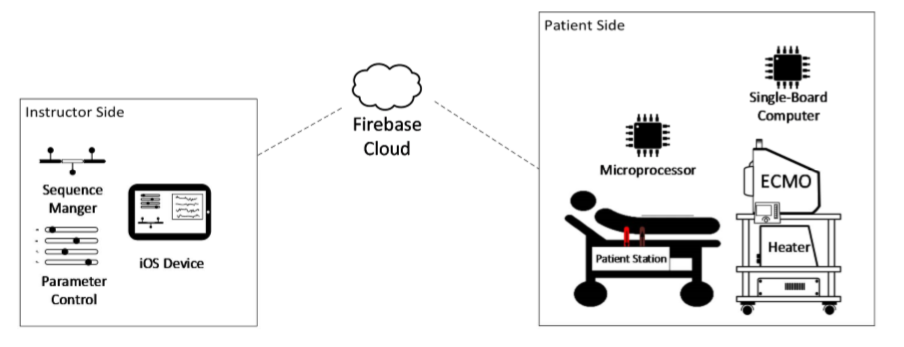


Figure 2. Kiến trúc hệ thống

### Mô tả

Hệ thống được triển khai gồm 3 phần chính: Cloud server trên nền tảng Firebase (bao gồm dịch vụ Firebase RD, Cloud Function và Firebase Auth), ứng dụng di động và sensor node (esp8266). Trong đó Firebase phụ trách như một trung tâm CSDL để nhận và đồng bộ dữ liệu cho toàn hệ thống, các client sẽ nhận dữ liệu theo thời gian thực từ Firebase RD. Đồng thời các sensor node sẽ gửi dữ liệu thu thập được lên Firebase RD thông qua thiết bị Esp8266 với module wifi được tích hợp sẵn của mạch này. Hệ thống hoạt động trên mạng Internet và có thể đồng bộ dữ liệu hoàn toàn theo thời gian thực

### Chức năng chính của từng thành phần

#### Nền tảng Firebase

Là server chính với nhiệm vụ xử lý các yêu cầu của người dùng, tiếp nhận và xử lý các dữ liệu thu thập được từ hệ thống cảm biến, xử lý cạnh tranh trong quá trình dữ liệu được cập nhật đồng thời từ nhiều cảm biến hoặc người dùng khác nhau

* Firebase RD

Firebase RD đóng vai trò là trung tâm CSDL thực hiện chức năng lưu trữ CSDL cho hệ thống và cho phép thao tác truy vấn, thay đổi cơ sở dữ liệu theo thời gian thực khi có kết nối với Internet. Nếu không có kết nối, các thay đổi sẽ được lưu trữ cục bộ và đồng bộ ngay sau khi bắt được kết nối

* Cloud Function

Cloud function platform đóng vai trò theo dõi tình hình dữ liệu được cập nhật và thiết lập ngưỡng dữ liệu để đưa ra cảnh báo cho người dùng thông qua việc gửi thông báo đến các mobile client trong các trường hợp cần thiết, đồng thời ghi nhận lại các sự kiện xảy ra vào CSDL để người nhân viên có thể truy xuất lại thông tin khi cần

* Firebase Auth

Trong đề tài, nhóm sử dụng dịch vụ này để xác thực người dùng nhằm đảm bảo tính bảo mật cho hệ thống, đảm bảo chỉ có người có quyền hạn mới được truy cập vào nội dung tương ứng. Dịch vụ này cũng đồng thời xử lý các thao tác nhân viên tài khoản như thêm tài khoản hoặc xử lý yêu cầu thay đổi mật khẩu từ người dùng.

#### Ứng dụng di động

Có 2 nhóm đối tượng chính tương tác với ứng dụng di động là người dùng thông thường và sensor node.

Đối với người dùng thông thường, ứng dụng hỗ trợ các thao tác sử dụng thông thường như đăng nhập, xem và nhân viên thông tin. Cụ thể, ứng dụng hiện đang làm việc trên 2 phân loại người dùng với các thao tác như sau:

* Nhân viên nhân viên: xác thực tài khoản, cho phép người dùng tương tác với hệ thống, thực hiện các thao tác theo dõi, nhân viên dữ liệu và xử lý tình huống
* Tài xế: xác thực tài khoản, cập nhật và thao tác trên thông tin hành trình mà tài xế phụ trách trong quyền hạn cho phép

Đối tượng thứ 2 của ứng dụng di động là hệ thống sensor node. Đối với sensor node, ứng dụng chủ yếu nhận dữ liệu mà sensor node gửi đi (thông qua Cloud server trung gian là Firebase) để hiển thị cho người dùng thông thường. Ngoài ra, ứng dụng cũng hỗ trợ chức năng thiết lập cấu hình wifi cho sensor node để tài xế hoặc người cấu hình có thể dễ dàng thiết lập kết nối wifi cho mạch Esp8266 mà không cần kiến thức chuyên sâu.

#### Nút cảm biến

Các nút cảm biến (sensor node) có nhiệm vụ thu thập dữ liệu, cụ thể là dữ liệu về nồng độ cồn trong quá trình tài xế lái xe. Ngoài ra, sensor node cũng chịu trách nhiệm xử lý một số tác vụ khi phát hiện nồng độ vượt quá mức quy định như tạm thời khóa hệ thống khởi động xe hoặc reo lên chuông cảnh báo tùy vào từng tình huống.

## Thiết kế ứng dụng di động

## Thiết kế nút cảm biến

### Giải pháp phần cứng

* Sử dụng vi xử lý Esp8266
* Sử dụng cảm biến MQ3 để thu thập chỉ số nồng độ cồn
* Điều khiển relay cho các thao tác kích hoạt/ ngắt hệ thống khởi động xe và chuông
* Sử dụng mô hình xe điện đồ chơi để thí nghiệm

### Giải pháp phần mềm

Sử dụng Arduino IDE để lập trình và thư viện <firebase???> để giao tiếp với Firebase RD. Thực hiện cấu hình wifi qua chức năng SmartConfig

### Các loại thiết bị sử dụng

#### Vi xử lý Esp8266

* Mô tả
* Chức năng
* Thông số kỹ thuật

#### Cảm biến nồng độ cồn MQ3

* Mô tả
* Chức năng
* Thông số kỹ thuật

#### Relay???

* Mô tả
* Chức năng
* Thông số kỹ thuật

### Mô hình kết nối

#### Chức năng thu thập dữ liệu

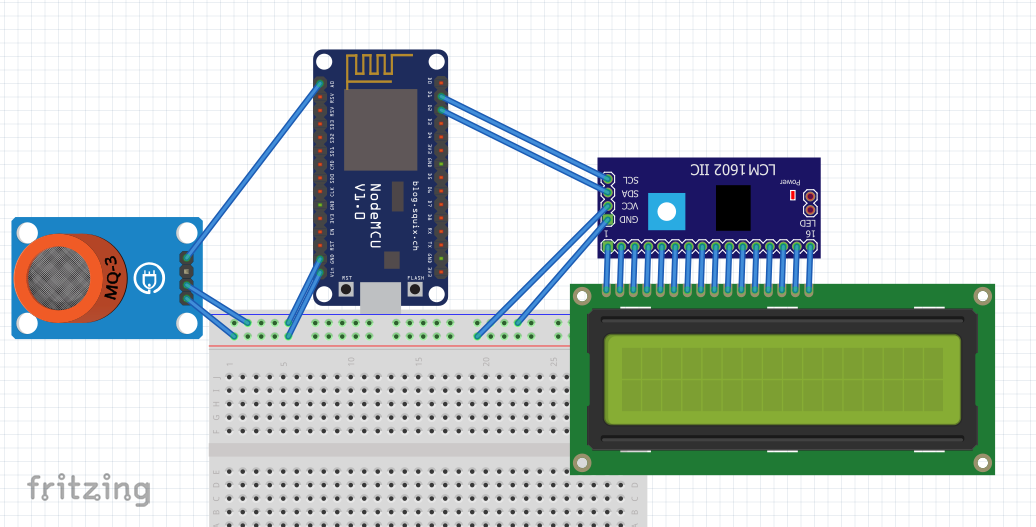


Figure 3. Mô hình kết nối chức năng thu thập dữ liệu

* Chức năng
* Thu thập giá trị nồng độ cồn trong không khí. Giá trị dữ liệu thu thập được là giá trị analog, giá trị này sẽ được chuyển đổi xấp xỉ về giá trị Blood Alcohol Content (BAC) - giá trị nồng độ cồn trong máu
* Hiển thị giá trị BAC lên màn hình
* Dùng làm dữ liệu cho chức năng kiểm tra trước khi khởi động và trong quá trình hoạt động
* Lấy giờ từ hệ thống NTP server và gửi timestamp cùng với giá trị nồng độ cồn để ghi nhận thời gian. Timestamp được ghi nhận dưới dạng milliseconds since epoch nên không bị ảnh hưởng bởi múi giờ mà tài xế hoạt động
* Công nghệ có liên quan
* Network Time Protocol - NTP là thuật toán phần mềm giữ cho các máy tính và các thiết bị công nghệ khác có thể đồng bộ hóa thời gian với nhau
* Máy chủ thời gian - NTP server là thiết bị được sử dụng để nhận biết yêu cầu đồng bộ thời gian và phân phối tín hiệu thông tin thời gian.

<https://cuongquach.com/ntp-server-la-gi-tong-quan-ve-dich-vu-network-time-protocol.html>

* Epoch time là số giây đếm được kể từ ngày 01/01/1970 (UTC) 00:00:00

#### Chức năng kiểm tra khi khởi động xe

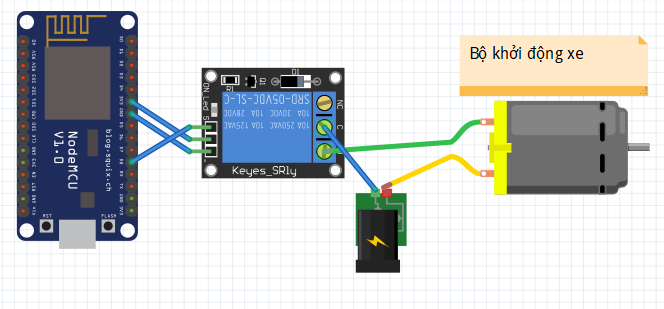


Figure 4. Mô hình kết nối chức năng kiểm tra khi khởi động xe

* Chức năng

Kiểm tra nồng độ cồn của tài xế có hợp lệ không trong chu kỳ 30 giây liên tục. Hệ thống khởi động xe mặc định ở trạng thái khóa do kết nối với cổng Normal Opened (cổng thường mở) của relay.

* Nếu hợp lệ: cho phép khởi động
* Không hợp lệ: thực hiện lại chu kỳ kiểm tra

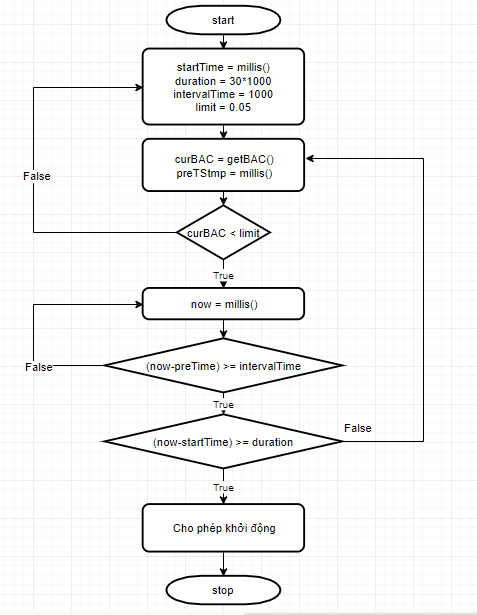


Figure 5. Lưu đồ hoạt động của chức năng kiểm tra khi khởi động xe

#### Chức năng kiểm tra trong quá trình chạy xe

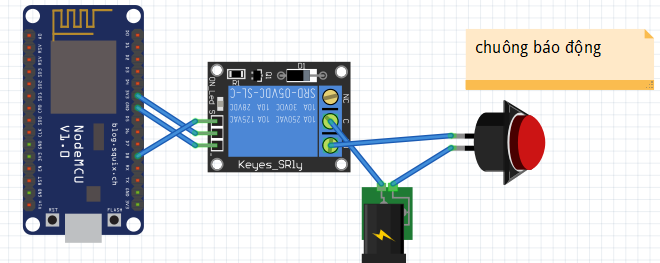


Figure 6. Mô hình kết nối chức năng kiểm tra trong quá trình chạy xe

* Chức năng
* Kiểm tra và cập nhật giá trị BAC theo thời gian thực lên hệ thống
* Kích hoạt chuông báo động cảnh báo người đi đường khi phát hiện vượt mức nồng độ cồn liên tục trong thời gian hơn 30 giây

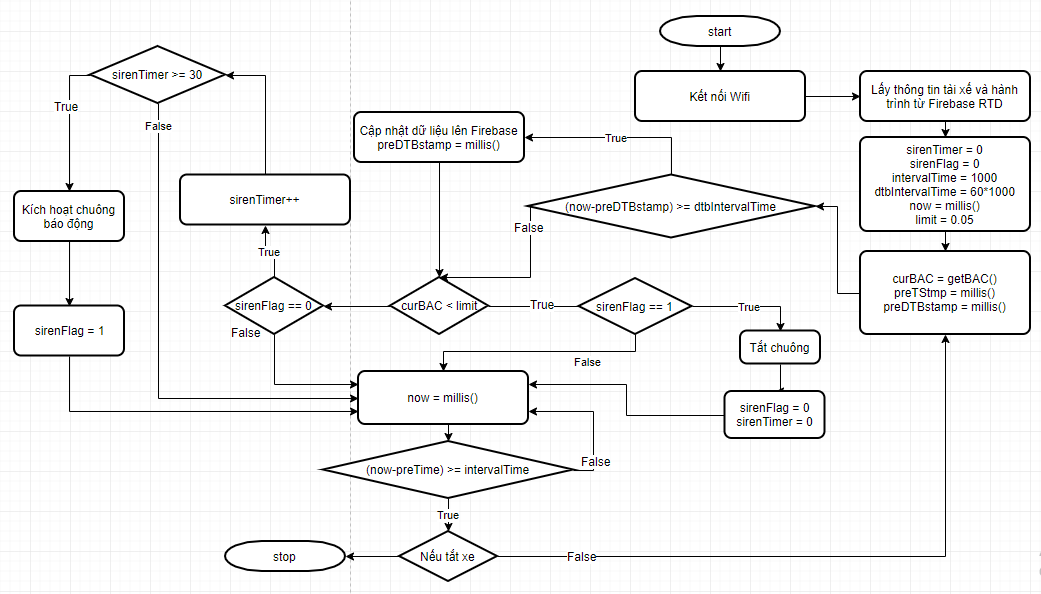


Figure 7. Lưu đồ hoạt động của chức năng kiểm tra trong quá trình chạy xe

#### Chức năng gửi dữ liệu lên Firebase RD

* Chức năng

Cập nhật giá trị nồng độ cồn theo thời gian lên CSDL Firebase RD

* Công nghệ có liên quan

Để kết nối với Firebase RD server, ta sử dụng thư viện FirebaseArduino <https://firebase-arduino.readthedocs.io/en/latest/>

* Hàm *void begin(const String &host, const String &auth = "")* : khởi tạo kết nối đến Firebase RD server với các thông tin được truyền vào tham số

Table 4. Bảng tham số hàm begin

|  |  |  |
| --- | --- | --- |
| Tên tham số | Ý nghĩa tham số | Ví dụ |
| host | địa chỉ máy chủ CSDL cần kết nối tới, thường có dạng *X.firebaseio.com* | alcohol-management-7fe33 .firebaseio.com |
| auth | thông tin truy cập xác thực vào CSDL, có thể là mật mã hoặc mã token | JiRTwBs3EL5mToelvXVOOP F3G8rX6SyleR57zxG7 |

* Hàm *void setFloat(const String &path, int value)*: ghi giá trị số thực (float) vào địa chỉ được truyền vào tham số path.

Table 5. Bảng tham số hàm setFloat

|  |  |  |
| --- | --- | --- |
| Tên tham số | Ý nghĩa tham số | Ví dụ |
| path | địa chỉ đường dẫn của node mà ta muốn ghi giá trị vào | /driver/TX0001/alcoholVal |
| value | giá trị muốn ghi vào | 0.15 |

#### Chức năng nhận dữ liệu từ Firebase RD

* Chức năng

Nhận dữ liệu tài xế và hành trình đang làm việc tương ứng với sensor node đang hoạt động

* Công nghệ có liên quan

Để kết nối với Firebase RD server, ta sử dụng thư viện FirebaseArduino

* Hàm *String getString(const String &path):* đọc giá trị chuỗi ký tự (String) ở địa chỉ được truyền vào tham số path. Hàm trả về giá trị chuỗi ký tự.

Table 6. Bảng tham số hàm getString

|  |  |  |
| --- | --- | --- |
| Tên tham số | Ý nghĩa tham số | Ví dụ |
| path | địa chỉ đường dẫn của node mà ta muốn đọc giá trị vào | /vehicles/V0001/ onWorking/tID |

## Kiểm thử

### Mục tiêu

* Kiểm tra các lỗi có thể xảy ra khi phát hành ứng dụng
* Đảm bảo tính hoàn thiện của sản phẩm
* Đảm bảo sản phẩm thực hiện đúng chức năng mong muốn

### Phạm vi

Kiểm thử các trường hợp tương ứng với các chức năng của hệ thống

#### Các chức năng sẽ được kiểm thử

* Xác thực tài khoản, thay đổi mật khẩu
* Thêm, sửa dữ liệu
* Chức năng thông báo
* Chức năng kiểm tra khi khởi động xe của nút cảm biến
* Chức năng kiểm tra quá trình lái xe của nút cảm biến
* Chức năng cập nhật dữ liệu của nút cảm biến

#### Các chức năng không được kiểm thử

* Trường hợp không có kết nối wifi
* Một số chức năng như xóa dữ liệu, phân công tài xế, chỉ định phương tiện do tương đồng với chức năng them, sửa dữ liệu
* Kiểm thử tải, kiểm thử hiệu năng

### Cách tiếp cận

Kiểm thử bằng phương pháp kiểm thử hộp đen, ghi nhận kịch bản vả kết quả đối với từng tình huống kiểm thử

#### Tiêu chí kiểm thử thành công/ thất bại

* Tiêu chí kiểm thử thành công: kết quả kiểm thử đạt được yêu cầu trong tài liệu đặc tả
* Tiêu chí kiểm thử thất bại: kết quả kiểm thử thực tế khác với kết quả mong đợi đã được ghi nhận trong tài liệu đặc tả

#### Tiêu chí đình chỉ và yêu cầu bắt đầu lại

* Tiêu chỉ đình chỉ: dừng thực hiện công việc khi dịch vụ phát sinh lỗi/ kiểm thử thất bại
* Yêu cầu bắt đầu lại: khi đã sửa được lỗi phát sinh ở dịch vụ bị đình chỉ

### Nhân viên kiểm thử

#### Kế hoạch kiểm thử

* Xây dựng trường hợp kiểm thử: viết các trường hợp kiểm thử cho mỗi dịch vụ cần kiểm thử
* Chuẩn bị môi trường kiểm thử: chuẩn bị các môi trường phần cứng, phân mềm cần thiết
* Cài đặt hệ thống kiểm thử: cài đặt dịch vụ lên môi trường kiểm thử
* Tiến hành kiểm thử: Thực hiện kiểm thử dịch vụ với các trường hợp đã lên kế hoạch
* Ghi nhận và đánh giá kết quả kiểm thử: ghi lại kết quả các trường hợp kiểm thử và đưa ra đánh giá
* Thực hiện kiểm thử lại nếu cần thiết

#### Môi trường kiểm thử

* Phần cứng:
* Bộ xử lý:???
* Ram:???
* Cấu hình mạng:????
* Phần mềm:
* Hệ điều hành: Android???

#### Trách nhiệm và quyền hạn

Table 7. Bảng phân công kiểm thử

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Họ tên | Công việc | | | | | |
| Nhân viên | Thiết kế | Chuẩn bị | Thực hiện | Chứng kiến | Kiểm tra |
| Nguyễn Bảo Ngọc |  | x |  | x |  | x |
| Đặng Quách Gia Bình | x |  | x | x | x | x |
| Bùi Xuân Huỳnh |  |  |  | x | x |  |
| Trần Phương Anh |  |  |  |  |  |  |

#### Giao tiếp giữa các nhóm liên quan

* Nút cảm biến (esp8266): do Đặng Quách Gia Bình phụ trách
* Ứng dụng di động: do Nguyễn Bảo Ngọc và Bùi Xuân Huỳnh phối hợp phụ trách

### Các trường hợp kiểm thử

#### Xác thực tài khoản, thay đổi mật khẩu

* Mục tiêu: đảm bảo người dùng có quyền truy cập có thể truy cập thành công và ngăn chặn các tình huống truy cập sai quyền hạn
* Các yêu cầu, thủ tục đặc biệt: kiểm thử viên phải có tài khoản được cấp quyền truy cập vào hệ thống

|  |  |  |
| --- | --- | --- |
| Loại người dùng | Tài khoản | Mật khẩu |
| Nhân viên | nv0001@potatoes.manager.com | 123123 |
| Tài xế | tv0001@potatoes.driver.com | 123456 |

* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH1\_01 | Tên tài khoản sai, mật khẩu sai | Sai toàn bộ thông tin |
| TH1\_02 | Tên tài khoản sai, mật khẩu đúng | Sai tài khoản |
| TH1\_03 | Tên tài khoản đúng, mật khẩu sai | Sai mật khẩu |
| TH1\_04 | (tài khoản nhân viên) tên tài khoảng đúng, mật khẩu đúng | Đăng nhập đúng tài khoản nhân viên |
| TH1\_05 | (tài khoản nhân viên) tên tài khoảng đúng, mật khẩu đúng | Đăng nhập đúng tài khoản tài xế |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH1\_01 | Thông báo lỗi | Như mong đợi |  |
| TH1\_02 | Thông báo lỗi | Như mong đợi |  |
| TH1\_03 | Thông báo lỗi | Như mong đợi |  |
| TH1\_04 | Đăng nhập thành công vào màn hình nhân viên | Như mong đợi |  |
| TH1\_05 | Đăng nhập thành công vào màn hình tài xế | Như mong đợi |  |

#### Thêm, sửa dữ liệu

* Mục tiêu: đảm bảo người dùng nhân viên thực hiện thao tác hợp lệ trên dữ liệu
* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH2\_01 | Nhập thiếu thông tin | Thiếu một hoặc nhiều thông tin |
| TH2\_02 | Không nhập thông tin | Thiếu toàn bộ thông tin |
| TH2\_03 | Nhập tên có kí tự đặc biệt | Sai định dạng tên |
| TH2\_04 | Nhập sai định dạng số điện thoại |  |
| TH2\_05 | Nhập sai định dạng CMND/ mã căn cước công dân |  |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH2\_01 | Thông báo nhập thiếu thông tin | Như mong đợi |  |
| TH2\_02 | Thông báo nhập thiếu thông tin | Như mong đợi |  |
| TH2\_03 | Thông báo lỗi | Như mong đợi |  |
| TH2\_04 | Thông báo lỗi | Như mong đợi |  |
| TH2\_05 | Thông báo lỗi | Như mong đợi |  |

#### Chức năng thông báo

* Mục tiêu: đảm bảo thông báo hợp lý
* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH3\_01 | Liên tục phát hiện nồng độ cồn của cùng một tài xế | Kiểm tra tính năng tạm ngừng thông báo, tránh tình trạng thông báo liên tục gây khó khăn cho việc sử dụng |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH3\_01 | Tạm ngừng thông báo của tài xế vi phạm, sau thời gian chờ 10 phút gửi thông báo mới nếu có | Như mong đợi |  |

#### Chức năng kiểm tra khi khởi động xe của nút cảm biến

* Mục tiêu: đảm bảo quá trình kiểm tra của nút cảm biến khi tài xế khởi động xe
* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH4\_01 | Chỉ số nồng độ cồn luôn ở mức quy định (BAC dưới 0.05) trong 30 giây liên tục |  |
| TH4\_02 | Phát hiện nồng độ cồn vượt mức quy định (BAC từ 0.05 trở lên) |  |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH4\_01 | Kích relay, cho phép khởi động xe | Như mong đợi |  |
| TH4\_02 | Bắt đầu lại quy trình kiểm tra (30 giây) - trong quá trình này relay vẫn ở trạng thái mở - không cho phép dòng điện chạy qua để tài xế không thể khởi động xe | Như mong đợi |  |

#### Chức năng kiểm tra trong quá trình lái xe của nút cảm biến

* Mục tiêu: đảm bảo quá trình kiểm tra của nút cảm biến khi tài xế lái xe
* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH5\_01 | Chỉ số nồng độ cồn ở mức chấp nhận (BAC dưới 0.04) |  |
| TH5\_02 | Chỉ số nồng độ cồn ở mức gần báo động (BAC từ 0.04 đến dưới 0.05) |  |
| TH5\_03 | Phát hiện nồng độ cồn vượt mức quy định (BAC từ 0.05 trở lên) trong thời gian ngắn hơn 30 giây khi chuông báo động đang tắt |  |
| TH5\_04 | Phát hiện nồng độ cồn vượt mức quy định (BAC từ 0.05 trở lên) trong thời gian từ 30 giây liên tục trở lên khi chuông báo động đang tắt |  |
| TH5\_05 | Phát hiện nồng độ cồn vượt mức quy định (BAC từ 0.05 trở lên) khi chuông báo động đang kêu |  |
| TH5\_06 | Nồng độ cồn trở về bình thường (sau khi đã có tín hiệu vượt mức trước đó) |  |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH5\_01 | Báo hiệu đèn xanh (tín hiệu hợp lệ) cho tài xế, cập nhật dữ liệu lên CSDL định kỳ | Như mong đợi |  |
| TH5\_02 | Báo hiệu đèn vàng (tín hiệu chú ý) cho tài xế, cập nhật dữ liệu lên CSDL định kỳ | Như mong đợi |  |
| TH5\_03 | Bắt đầu bộ đếm kích hoạt chuông, cập nhật dữ liệu lên CSDL định kỳ | Như mong đợi | Bộ đếm kích chuông để kiểm tra nồng độ cồn có vượt mức hơn 30 giây liên tục |
| TH5\_04 | Khởi động chuông cảnh báo, cập nhật dữ liệu lên CSDL định kỳ (CSDL sẽ kích hoạt thông báo cho người nhân viên) | Như mong đợi |  |
| TH5\_05 | Giữ chuông cảnh báo, cập nhật CSDL định kỳ (CSDL sẽ kích hoạt thông báo cho người nhân viên) | Như mong đợi |  |
| TH5\_06 | Đặt bộ đếm kích hoạt chuông về giá trị 0 hoặc tắt chuông báo động nếu chuông đang reo | Như mong đợi |  |

#### Chức năng cập nhật dữ liệu của nút cảm biến

* Mục tiêu: đảm bảo nút cảm biến cập nhật đúng dữ liệu
* Trường hợp

|  |  |  |
| --- | --- | --- |
| Mã số trường hợp | Nội dung kiểm thử | Ghi chú |
| TH6\_01 | Chỉ số nồng độ cồn luôn ở mức quy định (BAC dưới 0.05) khi tài xế lái xe |  |
| TH6\_02 | Phát hiện nồng độ cồn vượt mức quy định (BAC từ 0.05 trở lên) khi tài xế lai xe |  |

* Kết quả

|  |  |  |  |
| --- | --- | --- | --- |
| Mã số trường hợp | Kết quả mong đợi | Kết quả thực tế | Ghi chú |
| TH6\_01 | +Cập nhật dữ liệu định kỳ lên mục giá trị nồng độ hiện tại của tài xế  +Lưu vào hành trình kèm thời gian | Như mong đợi |  |
| TH6\_02 | +Cập nhật dữ liệu định kỳ lên mục giá trị nồng độ hiện tại của tài xế  +Lưu vào hành trình kèm thời gian  +Cập nhật mục thông báo để kích hoạt chức năng thông báo | Như mong đợi |  |

#### 

# Kết quả đạt được

## Kết quả

## Ưu

## Nhược

1. Kết luận
   1. Kết luận
   2. Kiến nghị
2. Tài liệu tham khảo