### 

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**VIỆN KỸ THUẬT - CÔNG NGHỆ**

**NGÀNH KỸ THUẬT PHẦN MỀM**

…………..o0o…………..



**BÁO CÁO TIỂU LUẬN**

**TIỂU LUẬN MÔN: PHÁT TRIỂN PHẦN MỀM NHÚNG**

**HỆ THỐNG CẢNH BÁO HỎA HOẠN VÀ KHÍ ĐỘC**

Lê Thanh Hiệp 1824801030271

Phạm Thị Ngọc Hân 1824801030225

Hồ Ngọc Giả 1424801030021

GVHD: Trịnh Quốc Thanh

***Bình Dương, ngày 27 tháng 07 năm 2021***

# LỜI CẢM ƠN

Để hoàn thành đề tài nghiên cứu này, lời đầu tiên cho phép nhóm chúng em được gửi lời cảm ơn chân thành đến Viện Kỹ Thuật – Công nghệ, những thầy(cô) đã tận tình hướng dẫn, trang bị cho nhóm sinh viên những kiến thức nền tảng và kiến thức chuyên ngành quan trọng, giúp nhóm sinh viên có được cơ sở lý thuyết vững vàng và đã luôn tạo điều kiện giúp đỡ tốt nhất cho nhóm sinh viên trong quá trình học tập và nghiên cứu.

Đặc biệt, nhóm chúng em xin chân thành cảm ơn thầy Trịnh Quốc Thanh đã tận tình giúp đỡ, đưa ra những định hướng nghiên cứu cũng như hướng giải quyết một số vấn đề để nhóm có thể thực hiện tốt đề tài. Trong thời gian làm việc với thầy, nhóm chúng em đã không ngừng tiếp thu thêm nhiều kiến thức được chỉ dạy từ thầy, luôn thể hiện một thái độ nghiên cứu nghiêm túc, hiệu quả và đây cũng là điều rất cần thiết trong quá trình học tập và làm việc sau này đối với nhóm.

Mặc dù đã cố gắng hết sức, xong do điều kiện thời gian và kinh nghiệm thực tế của nhóm nghiên cứu còn ít, cho nên đề tài không thể tránh khỏi thiếu sót. Vì vậy, nhóm chúng em rất mong nhận được sự đóng góp ý kiến của quý thầy(cô) giáo.

Xin chân thành cảm ơn !

*Bình Dương, ngày 27 tháng 07 năm 2021.*

**Sinh viên thực hiện**

Lê Thanh Hiệp

Phạm Thị Ngọc Hân

Hồ Ngọc Giả

# MỤC LỤC

[1](#_Toc78366122)

[LỜI CẢM ƠN 1](#_Toc78366123)

[MỤC LỤC 2](#_Toc78366124)

[DANH SÁCH HÌNH MINH HỌA 5](#_Toc78366125)

[DANH SÁCH BẢNG SỐ LIỆU 7](#_Toc78366126)

[TÓM TẮT ĐỀ TÀI 8](#_Toc78366127)

[CHƯƠNG I: GIỚI THIỆU 9](#_Toc78366128)

[1.1 Tổng quan tình hình nghiên cứu: 9](#_Toc78366129)

[1.2 Mục tiêu: 10](#_Toc78366130)

[1.3 Giới hạn đề tài: 10](#_Toc78366131)

[1.4 Phương pháp nghiên cứu: 11](#_Toc78366132)

[1.5 Đặt vấn đề: 11](#_Toc78366133)

[1.6 Nội dung thực hiện: 12](#_Toc78366134)

[CHƯƠNG II: CƠ SỞ LÝ THUYẾT 13](#_Toc78366135)

[2.1 Giới thiệu về hệ thống IoT: 13](#_Toc78366136)

[2.2 Tìm hiểu Kit Wi-fi ESP8266 NodeMCU Lua CP2102: 13](#_Toc78366137)

[**2.2.1 Giới thiệu chung:** 13](#_Toc78366138)

[**2.2.2 Thông số kỹ thuật:** 14](#_Toc78366139)

[**2.2.3 Sơ đồ chân Kit Wi-fi ESP8266 NodeMCU Lua CP2102:** 15](#_Toc78366140)

[2.3 Tìm hiểu Chip Wifi SoC ESP8266: 16](#_Toc78366141)

[**2.3.1 Giới thiệu SoC ESP8266:** 16](#_Toc78366142)

[**2.3.2 Thông số phần cứng:** 18](#_Toc78366143)

[2.4 Cảm biến khí gas MQ-2: 18](#_Toc78366144)

[**2.4.1 Giới thiệu chung:** 18](#_Toc78366145)

[**2.4.2 Thông số kĩ thuật:** 19](#_Toc78366146)

[2.5 Cảm biến nhiệt độ và độ ẩm của môi trường DHT11: 20](#_Toc78366147)

[**2.5.1 Giới thiệu chung:** 20](#_Toc78366148)

[**2.5.2 Thông số kĩ thuật:** 21](#_Toc78366149)

[**2.5.3 Sơ đồ chân của Module:** 22](#_Toc78366150)

[2.6 Cảm biến chất lượng không khí MQ-135: 22](#_Toc78366151)

[**2.6.1 Giới thiệu chung:** 22](#_Toc78366152)

[**2.6.2 Thông số kĩ thuật:** 23](#_Toc78366153)

[**2.6.3 Sơ đồ bố trí và kích thước của Module MQ-135:** 24](#_Toc78366154)

[2.7 Phần mềm: 24](#_Toc78366155)

[**2.7.1 Arduino IDE:** 24](#_Toc78366156)

[**2.7.2 Android Studio:** 26](#_Toc78366157)

[2.8 Chuẩn Truyền Dữ Liệu Nối Tiếp UART: 26](#_Toc78366158)

[**2.8.1 Quá trình truyền dữ liệu UART:** 27](#_Toc78366159)

[**2.8.2 Thông số chuẩn truyền UART:** 27](#_Toc78366160)

[2.9 Sử dụng Website firebase.google.com: 29](#_Toc78366161)

[**2.9.1 Giới thiệu:** 29](#_Toc78366162)

[**2.9.2 Sử dụng nền tảng Firebase:** 30](#_Toc78366163)

[CHƯƠNG III: THIẾT KẾ HỆ THỐNG 33](#_Toc78366164)

[3.1 Yêu cầu của hệ thống: 33](#_Toc78366165)

[3.2 Thiết kế phần cứng: 33](#_Toc78366166)

[**3.2.1 Sơ đồ khối hệ thống và chức năng:** 33](#_Toc78366167)

[**3.2.2 Sơ đồ nguyên lý hệ thống:** 34](#_Toc78366168)

[**3.2.3 Lưu đồ giải thuật:** 35](#_Toc78366169)

[**3.2.4 Tính toán các khối:** 36](#_Toc78366170)

[3.3 Thiết kế phần mềm: 37](#_Toc78366171)

[**3.3.1 Thiết kế kiến trúc:** 37](#_Toc78366172)

[**3.3.2 Thiết kế giao diện:** 39](#_Toc78366173)

[**3.3.3 Thiết kế dữ liệu:** 44](#_Toc78366174)

[**3.3.4 Thiết kế thuật toán:** 44](#_Toc78366175)

[3.4 Hướng dẫn sử dụng và thao tác: 46](#_Toc78366176)

[**3.4.1 Hướng dẫn sử dụng:** 46](#_Toc78366177)

[**3.4.2 Quy trình thao tác:** 47](#_Toc78366178)

[CHƯƠNG IV: KẾT QUẢ THỰC HIỆN 48](#_Toc78366179)

[4.1 Phần cứng: 48](#_Toc78366180)

[**4.1.1 Thi công boarb mạch hệ thống:** 48](#_Toc78366181)

[**4.1.2 Thi công hộp bảo vệ:** 49](#_Toc78366182)

[4.2 Giao diện phần mềm: 52](#_Toc78366183)

[**4.2.1 Hệ thống đang hoạt động không có sự cố rò rỉ khí gas:** 52](#_Toc78366184)

[**4.2.2 Hệ thống đang hoạt động có sự cố rò rỉ khí gas:** 54](#_Toc78366185)

[CHƯƠNG V: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 56](#_Toc78366186)

[5.1 Kết luận: 56](#_Toc78366187)

[5.2 Các điểm hạn chế của hệ thống: 56](#_Toc78366188)

[5.3 Hướng phát triển: 56](#_Toc78366189)

[TÀI LIỆU THAM KHẢO 58](#_Toc78366190)

[1. Sách tham khảo: 58](#_Toc78366191)

[2. Trang Web tham khảo: 58](#_Toc78366192)

[PHỤ LỤC 59](#_Toc78366193)

[1. Phân chia công việc trong nhóm: 59](#_Toc78366194)

[2. Chương trình Phần cứng (Arduino IDE): 59](#_Toc78366195)

[3. Chương trình Phần mềm (App): 63](#_Toc78366196)

[4. Chương trình cho Firebase.google.com: 87](#_Toc78366197)

# DANH SÁCH HÌNH MINH HỌA

**Hình 2.1: Kit Wi-fi ESP8266 NodeMCU Lua CP2102. 14**

**Hình 2.2: Sơ đồ chân kit Wifi ESP8266 NodeMCU. 15**

**Hình 2.3: Sơ đồ chân & sơ đồ khối ESP8266EX. 17**

**Hình 2.4: Cảm biến khí MQ-2. 19**

**Hình 2.5: Module cảm biến khí Gas MQ-2. 20**

**Hình 2.6: Module cảm biến nhiệt độ, độ ẩm DHT11. 21**

**Hình 2.7: Sơ đồ chân Module cảm biến nhiệt độ, độ ẩm DHT11. 22**

**Hình 2.8:Cảm biến kiểm tra chất lượng không khí MQ-135. 23**

**Hình 2.9: Sơ đồ bố trí và kích thước của Module MQ-135. 24**

**Hình 2.10: Sơ đồ kết nối của Module MQ-135. 24**

**Hình 2.11: Giao diện phần mềm Arduino IDE. 25**

**Hình 2.12: Giao diện Start khi mở phần mềm Android Studio. 26**

**Hình 2.13: Cấu trúc cơ bản của 1 frame dữ liệu. 27**

**Hình 2.14: Trang chủ Firebase sau khi Sign In thành công. 30**

**Hình 2.15: Project đã Add thành công. 31**

**Hình 2.16: Giao diện Trang Firebase Realtime Database. 31**

**Hình 2.17: Lấy mã để truyền data. 32**

**Hình 2.18: Dữ liệu được upload lên. 32**

**Hình 3.1: Sơ đồ khối hệ thống. 33**

**Hình 3.2: Sơ đồ nguyên lý hệ thống. 34**

**Hình 3.3: Sơ đồ giải thuật phần cứng. 35**

**Hình 3.4: Mô hình kiến trúc phần mềm. 37**

**Hình 3.5: Mô hình Client – Server. 38**

**Hình 3.6: Giao diện Đăng nhập tài khoản để sử dụng phần mềm. 39**

**Hình 3.7: Giao diện Đăng kí tài khoản để sử dụng phần mềm. 40**

**Hình 3.8: Giao diện Thông tin tài khoản cá nhân. 41**

**Hình 3.9: Giao diện Quản lý hệ thống. 42**

**Hình 3.10: Giao diện Thông báo giá trị các thông số của cảm biến. 43**

**Hình 3.11: Cấu trúc dữ liệu. 44**

**Hình 3.12: Lưu đồ giải thuật phần mềm. 45**

**Hình 3.13: Quy trình thao tác hệ thống cảnh báo hoả hoạn và khí độc. 47**

**Hình 4.1: Cắm và kết nối các linh kiện lên Breadboard. 49**

**Hình 4.2: Băng keo trong. 49**

**Hình 4.3: Dao cắt giấy. 49**

**Hình 4.4: Hộp bảo vệ. 50**

**Hình 4.5: Hệ thống chưa mở. 50**

**Hình 4.6: Hệ thống đang hoạt động. 51**

**Hình 4.7: Thiết bị báo khu vực an toàn không phát hiện rò rỉ khí gas. 52**

**Hình 4.8: Dữ liệu trên Realtime Database khí gas không bị rò. 53**

**Hình 4.9: Thiết bị đang hoạt động khi có sự cố rò rỉ khí gas. 54**

**Hình 4.10: Dữ liệu trên Realtime Database phát hiện khí gas. 55**

**Hình 5.1: Mẫu hướng phát triển cho hệ thống. 57**

# DANH SÁCH BẢNG SỐ LIỆU

**Bảng 3.1: Bảng kết nôi dây các linh kiện điện tử. 34**

**Bảng 3.2: Bảng thống kê điện năng tiêu thụ của các linh kiện. 36**

**Bảng 4.1: Bảng linh kiện cần chuẩn bị. 48**

**Bảng 7.1. Bảng phân chia công việc trong nhóm. 59**

# TÓM TẮT ĐỀ TÀI

Thiết kế, xây dựng hệ thống cảnh báo hỏa hoạn và khí độc gồm một trạm xử lý và một trung tâm xử lý. Trạm xử lý có chức năng đo và kiểm tra các thông sô môi trường và gửi các thông số này cho trung tâm xử lý. Trung tâm xử lý sẽ hiển thị các thông số nhận được lên giao diện giám sát và thông báo những cảnh báo nếu có sự cố hỏa hoạn hoặc khí độc xảy ra thông qua kết nối Internet.

# CHƯƠNG I: GIỚI THIỆU

## 1.1 Tổng quan tình hình nghiên cứu:

* **Các đề tài nghiên cứu trong nước:**
* **Đề tài 1:**

- Tên đề tài: Hệ thống giám sát và báo động khí gas

- Tác giả: Đoàn Thanh Đủ

- Ngày viết: 07/2018

- Nội dung: Thi công hệ thống giám sát khí gas tại các trạm phát hiện rò khí gas theo từng khu vực hiển thị lên LCD, truyền nhận dữ liệu bằng sóng RF. Dùng arduino UNO cho hệ thống con, arduino Mega cho hệ thống chính điều khiển. Sử dụng loại cảm biến khí gas MQ-2, module SIM900A mini để cảnh báo từ xa, module ESP8266 Node MCU để truyền dữ liệu lên website. Mạch nguồn tự thiết kế để đảm bảo dòng áp ổn định cho toàn bộ hệ thống.

* **Đề tài 2:**

- Tên đề tài: Thiết kế và thi công hệ thống cảnh báo, phòng chống hỏa hoạn và rò rỉ khí gas.

- Tác giả: Trần Minh Tâm, Đỗ Thị Huê

- Ngày viết: 6/2019

- Nội dung: Thi công hệ thống thu thập dữ liệu từ cảm biến lửa, nhiệt độ, khí gas hiển thị lên LCD và trang web Thingspeak.com để theo dõi nồng độ khí gas, nhiệt độ trong phòng. Nếu phát hiện có gas, lửa hay nhiệt độ quá cao sẽ báo động bằng cách gọi, gửi tin nhắn cho người sử dụng và mở quạt thông gió nhằm làm giảm nồng độ khí gas trong không khí, kích hoạt máy bơm hoạt động dập tắt lửa. Sử dụng vi điều khiển Arduino Nano làm khối xử lí trung tâm, các trạm giám sát truyền nhận thông tin bằng sóng RF. Sử dụng cảm biến phát hiện lửa flame sensor, cảm biến nhiệt độ DHT-22, cảm biến phát hiện khí gas MQ-2, động cơ bơm P385.

- Ngày viết: 21/05/2019

- Nội dung: Mô hình cảm biến khí gas có lưu đồ giải thuật rõ ràng, độ nhạy cao. Bên cạnh đó mô hình khá đơn giản, không có cảnh báo từ xa, chỉ có đèn cảnh báo.

- Ngày viết: 01/02/2018

- Nội dung: Hướng dẫn cụ thể cách kết nối phần cứng, phần code đơn giản nhưng tối ưu. Mô hình hiển thị giá trị thông số cảm biến chính xác trên màn hình LCD. Tuy nhiên, mô hình chỉ hướng dẫn kết nối cơ bản các thiết bị chứ chưa thật sự tối ưu, mạch không có cảnh báo chỉ có hiển thị thông số cảm biến.

## 1.2 Mục tiêu:

“Hệ thống cảnh báo hỏa hoạn và khí độc” sẽ phát hiện được các dấu hiệu của sự cháy nổ, khí nguy hiểm và vị trí của nó sau đó cảnh báo cho chúng ta kịp thời phòng tránh và đưa ra các phương án xử lý hiệu quả nhất. Đề tài sử dụng vi điều khiển làm bộ xử lý trung tâm, các cảm biến nhiệt độ và độ ẩm, cảm biến phát hiện khí độc giám sát trực tiếp, phát hiện ra dấu hiệu của hỏa hoạn và khí độc. Module Wi-Fi nhận tín hiệu từ trung tâm sẽ gửi dữ liệu lên App giám sát của người quản lý để kịp thời thông báo cho mọi người trong trường hợp có sự cố hỏa hoạn hoặc khí độc xảy ra.

## 1.3 Giới hạn đề tài:

- Sử dụng vi điều khiển làm trung tâm điều khiển.

- Thiết bị đầu vào: nhận thông tin nơi xảy ra sự cháy và truyền tín hiệu đến

trung tâm báo cháy.

+ Cảm biến khói giám sát trực tiếp phát hiện ra dấu hiệu khói.

+ Cảm biến chất lượng không khí giám sát trực tiếp phát hiện ra dấu hiệu

của khí độc nguy hiểm.

+ Cảm biến nhiệt kiểm tra nhiệt độ và độ ẩm của môi trường trong phạm vi được bảo vệ, khi nhiệt độ của môi trường không thoả mãn những quy định thiết lập ban đầu, thì nó sẽ phát tín hiệu cảnh báo gửi về trung tâm xử lý.

- Thiết bị đầu ra: Nhận tín hiệu từ trung tâm báo cháy truyền đến.

+ Module Wi-Fi được tích hợp bên trong con vi điều khiển chính được dùng cho các ứng dụng cần kết nối, thu thập dữ liệu và điều khiển qua sóng Wifi, đặc biệt là các ứng dụng liên quan đến IoT.

## 1.4 Phương pháp nghiên cứu:

- Kết hợp giữa nghiên cứu lý thuyết và mô hình thực tế để làm rõ nội dung đề tài. Cụ thể như sau:

+ Thu thập, phân tích các tài liệu và thông tin liên quan đến đề tài.

+ Vận dụng những kiến thức cơ bản đã học về Phát triển phần mềm nhúng.

+ Tìm hiểu qua tài liệu internet và sách báo và nhu cầu đời sống xã hội.

+ Sử dụng phần mềm chuyên dụng (Arduino IDE) để thực hiện viết code và nạp code cho phần cứng, phần mềm Andoid Studio để viết ứng dụng (App) trên di dộng.

+ Tìm hiểu các đồ án có đề tài liên quan.

+ Sử dụng các phần mềm vẽ mô phỏng và lập trình.

## 1.5 Đặt vấn đề:

Ngày nay, cùng với những hiểm hoạ có thể xảy ra với con người thì hoả hoạn cũng là một trong những mối nguy hiểm mà con người cần đề phòng nhất. Hậu quả mà nó gây ra cho chúng ta là rất lớn và khó có thể lường được. Do đó vấn đề đặt ra ở đây là chúng ta cần cảnh giác cao về phòng cháy chữa cháy. Chúng ta cần trang bị đầy đủ những phương tiện phòng cháy chữa cháy để kịp thời xử lý khi có xự cố xảy ra. Bên cạnh đó là một hệ thống phát hiện và báo động kịp thời cũng rất quan trọng. Chỉ có những hệ thống báo cháy, chữa cháy được thiết kế đúng đắn, đầy đủ chức năng, ổn định và đạt tiêu chuẩn mới có thể đảm bảo cho cao ốc, nhà xưởng, căn hộ của mình một cách chắc chắn khỏi những rủi ro do hoả hoạn gây ra.

Vì vậy, “Hệ thống cảnh báo hỏa hoạn và khí độc” được thiết kế dựa trên việc sử dụng hệ thống vi điều khiển cho việc điều khiển và giám sát những trạng thái nhiệt độ và độ ẩm, khí độc để phát hiện, phòng chống, cảnh báo và hạn chế những thiệt hại về tài sản và tính mạng con người.

## 1.6 Nội dung thực hiện:

Ở đề tài “Hệ thống cảnh báo hỏa hoạn và khí độc” Nhóm chúng em tiến hành thực hiện nghiên cứu các nội dung như sau:

- Tìm hiểu các mô hình cảnh báo hiện có, những hạn chế của nó. Từ đó lựa chọn giải pháp cho hệ thống của nhóm phát triển.

- Thiết kế sơ đồ khối, sơ đồ nguyên lý và các lưu đồ giải thuật, lựa chọn linh kiện phù hợp cho hệ thống.

- Nghiên cứu phương pháp truyền nhận dữ liệu thông qua chip Wifi SoC ESP8266 hỗ trợ giao tiếp WiFi.

- Thiết kế ứng dụng chạy trên điện thoại có hệ điều hành Android. Gửi giá trị của cảm biến và trạng thái các ngõ ra lên ứng dụng thông qua module Wi-Fi ESP8266 Node MCU.

- Nghiên cứu Thiết kế Firebase Cloud dựa trên nền tảng Node.js, HTML để thực hiện quá trình truyền – nhận dữ liệu cho hệ thống thông qua module thu – phát WiFi Esp8266 Node MCU. Liên kết và đồng bộ hóa dữ liệu giữa ứng dụng và website.

- Nguyên cứu IDE - môi trường lập trình cho Arduino, phần mềm Android Studio - môi trường lập trình di động.

# CHƯƠNG II: CƠ SỞ LÝ THUYẾT

## 2.1 Giới thiệu về hệ thống IoT:

Internet Vạn Vật, hay cụ thể hơn là Mạng lưới vạn vật kết nối Internet hoặc là Mạng lưới thiết bị kết nối Internet (tiếng Anh: Internet of Things, viết tắt IoT) là một liên mạng, trong đó các thiết bị, phương tiện vận tải (được gọi là "thiết bị kết nối" và "thiết bị thông minh"), phòng ốc và các trang thiết bị khác được nhúng với các bộ phận điện tử, phần mềm, cảm biến, cơ cấu chấp hành cùng với khả năng kết nối mạng máy tính giúp cho các thiết bị này có thể thu thập và truyền tải dữ liệu.

Hệ thống IoT cho phép vật được cảm nhận hoặc được điều khiển từ xa thông qua hạ tầng mạng hiện hữu, tạo cơ hội cho thế giới thực được tích hợp trực tiếp hơn vào hệ thống điện toán, hệ quả là hiệu năng, độ tin cậy và lợi ích kinh tế được tăng cường bên cạnh việc giảm thiểu sự can dự của con người.

Khi IoT được gia tố cảm biến và cơ cấu chấp hành, công nghệ này trở thành một dạng thức của hệ thống ảo-thực với tính tổng quát cao hơn, bao gồm luôn cả những công nghệ như điện lưới thông minh, nhà máy điện ảo, nhà thông minh, vận tải thông minh và thành phố thông minh. Mỗi vật được nhận dạng riêng biệt trong hệ thống điện toán nhúng và có khả năng phối hợp với nhau trong cùng hạ tầng Internet hiện hữu.

Khi tự động hóa có kết nối internet được triển khai đại trà ra nhiều lãnh vực, IoT được dự báo sẽ tạo ra lượng dữ liệu lớn từ đa dạng nguồn, kéo theo sự cần thiết cho việc kết tập dữ liệu nhanh, gia tăng nhu cầu đánh chỉ mục, lưu trữ, và xử lý các dữ liệu này hiệu quả hơn. Internet Vạn Vật hiện nay là một trong các nền tảng của Thành phố Thông minh, và các Hệ thống Quản lý Năng lượng Thông minh.

## 2.2 Tìm hiểu Kit Wi-fi ESP8266 NodeMCU Lua CP2102:

### **2.2.1 Giới thiệu chung:**

Kit ESP8266 là kít phát triển dựa trên nền chíp Wifi SoC ESP8266 với thiết kế dễ dàng sửa dụng vì tích hợp sẵn mạch nạp sử dụng chíp CP2102 trên board. Bên trong ESP8266 có sẵn một lõi vi sử lý vì thế bạn có thể trực tiếp lập trình cho ESP8266 mà không cần thêm bất kì con vi sử lý nào nữa. Hiện tại có hai ngôn ngữ có thể lập trình cho ESP8266, sử dụng trực tiếp phần mềm IDE của Arduino để lập trình với bộ thư viện riêng hoặc sử dụng phần mềm node MCU.

Kit thu phát Wifi ESP8266 NodeMCU được dùng cho các ứng dụng cần kết nối, thu thập dữ liệu và điều khiển qua sóng Wifi, đặc biệt là các ứng dụng liên quan đến IoT.

Kit thu phát Wifi ESP8266 NodeMCU sử dụng chip nạp và giao tiếp UART mới và ổn định nhất là CP2102 có khả năng tự nhận Driver trên tất cả các hệ điều hành Window và Linux, đây là phiên bản nâng cấp từ các phiên bản sử dụng IC nạp giá rẻ CH340.



**Hình 2.1: Kit Wi-fi ESP8266 NodeMCU Lua CP2102.**

### **2.2.2 Thông số kỹ thuật:**

- Chip: ESP8266 Wifi SoC (ESP-12).

- Chip nạp và giao tiếp UART: CP2102.

- Bộ nhớ Flash: 4MB.

- Ăng-ten: ăng-ten PCB.

- Giao thức kết nối ngoại vi: UART / SPI / I2C / SDIO / GPIO / ADC / PWM.

- Wifi protocol: IEEE 802.11 b / g / n.

- Dải tần số: 2.4G ~ 2.5G (2400M ~ 2483.5M).

- Chế độ WIFI:Station / SoftAP / SoftAP+Station.

- Giao tiếp: Cable Micro USB.

- Hỗ trợ bảo mật: WPA/WPA2.

- Tích hợp: TCP/IP.

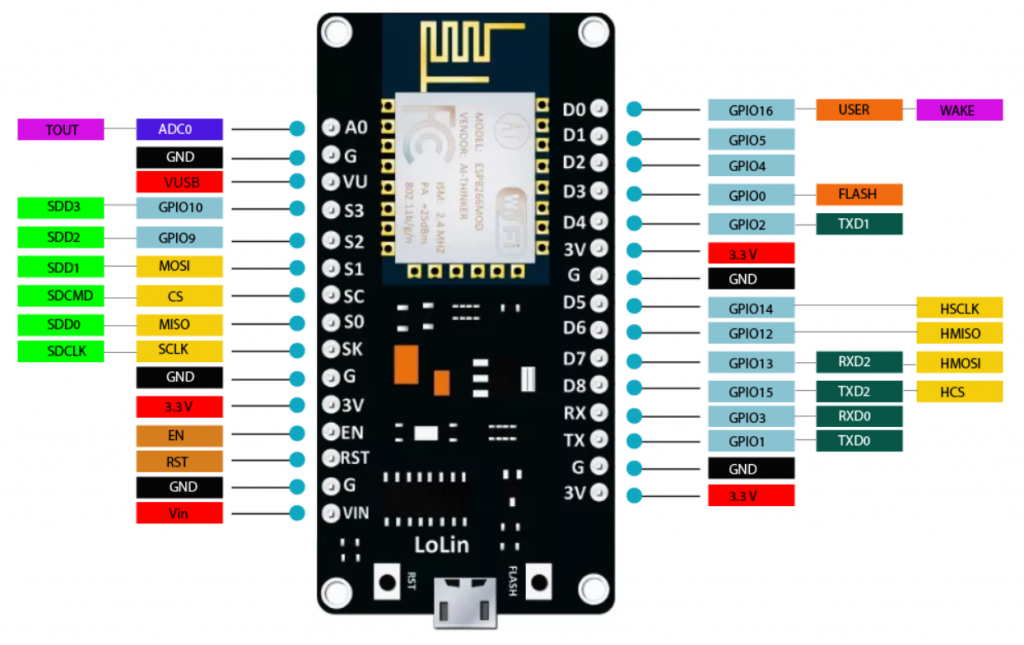
- Lập trình trên các ngôn ngữ: C/C++, Micropython, NodeMCU – Lua.

- Điện áp hoạt động: 5V, mức logic: 3V.

- Số chân I/O: 11 (tất cả các chân I/O đều có Interrupt/PWM/I2C/One-wire, trừ chân D0).

- Kích thước: 48.26mm x 25.4mm.

### **2.2.3 Sơ đồ chân Kit Wi-fi ESP8266 NodeMCU Lua CP2102:**



**Hình 2.2: Sơ đồ chân kit Wifi ESP8266 NodeMCU.**

Có tới 17 chân GPIO. Chúng có thể được gán cho các chức năng khác nhau. Mỗi GPIO có thể được cấu hình với kéo lên (ngoại trừ XPD\_DCDC, được cấu hình kéo xuống).

Các chân này được ghép với các chức năng khác như I2C, I2S, UART, PWM, Điều khiển từ xa IR,...

Tất cả các chân IO kỹ thuật số được bảo vệ khỏi điện áp quá mức với mạch đảo được kết nối giữa mạch dẫn và ground. Điện áp đảo thường là khoảng 6V, và điện áp giữ là 5.8V.

Các thiết bị đầu ra cũng được bảo vệ khỏi điện áp đảo ngược với diode.

## 2.3 Tìm hiểu Chip Wifi SoC ESP8266:

Các thông số môi trường sau khi được thu thập bởi trạm xử lý trung tâm sẽ được gửi về trung tâm xử lý. Để theo dõi và quản lý dữ liệu của trung tâm xử lý từ xa trên App Smartphone người dùng, nên các thông số sẽ được gửi thông qua sóng Wifi.

### **2.3.1 Giới thiệu SoC ESP8266:**

SoC ESP8266 là dòng chip tích hợp Wi-Fi 2.4Ghz có thể lập trình được, giá thành rẻ được sản xuất bởi một công ty bán dẫn Trung Quốc: Espressif Systems.

Được phát hành đầu tiên vào tháng 8 năm 2014, đóng gói đưa ra thị trường dạng Module ESP-01, được sản xuất bởi bên thứ 3: AI-Thinker. Có khả năng kết nối Internet qua mạng Wi-Fi một cách nhanh chóng và sử dụng rất ít linh kiện đi kèm. Với giá cả có thể nói là rất rẻ so với tính năng và khả năng ESP8266 có thể làm được.

ESP8266 có một cộng đồng các nhà phát triển trên thế giới rất lớn, cung cấp nhiều Module lập trình mã mở giúp nhiều người có thể tiếp cận và xây dựng ứng dụng rất nhanh.

Hiện nay tất cả các dòng chip ESP8266 trên thị trường đều mang nhãn ESP8266EX, là phiên bản nâng cấp của ESP8266.

Hiện tại có khá nhiều module khác nhau cho ESP8266 được sản xuất bởi công ty AI-Thinker. Đặc điểm khác nhau giữa các module này bao gồm:

- Loại anten sử dụng (PCB anten, chip anten hoặc gắn anten ngoài).

- Dung lượng của chip Flash SPI trên board.

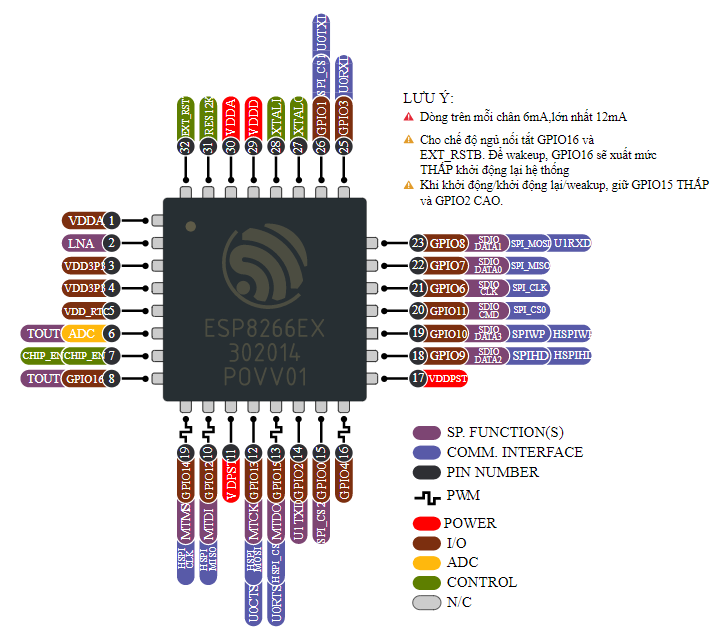
- Kích thước board của module.

- Có gắn khung nhôm chống nhiễu hay không.

- Số lượng pin GPIO đưa ra chân kết nối.

AI-Thinker sản xuất 14 loại module cho ESP từ module ESP-01 đến ESP-14.

Ở thị trường Việt Nam thì 3 module là ESP-01, ESP-07 và ESP-12 (loại ESP dùng cho kit ESP8266 MCU) khá phổ biến.

* **Sơ đồ chân ESP8266:**

**Hình 2.3: Sơ đồ chân & sơ đồ khối ESP8266EX.**

### **2.3.2 Thông số phần cứng:**

- 32-bit RISC CPU : Tensilica Xtensa LX106 running at 80 MHz.

- Hổ trợ Flash ngoài từ 512KiB đến 4MiB.

- 64KBytes RAM thực thi lệnh.

- 96KBytes RAM dữ liệu.

- 64KBytes boot ROM.

- Chuẩn wifi EEE 802.11 b/g/n, Wi-Fi 2.4 GHz Tích hợp TR switch, balun, LNA, khuếch đại công suất và matching network Hổ trợ WEP, WPA/WPA2, Open network.

- Tích hợp giao thức TCP/IP.

- Hổ trợ nhiều loại anten.

- 16 chân GPIO.

- Hổ trợ SDIO 2.0, UART, SPI, I²C, PWM, I²S với DMA.

- 1 ADC 10-bit.

- Dải nhiệt độ hoạt động rộng: -40C ~ 125C.

## 2.4 Cảm biến khí gas MQ-2:

### **2.4.1 Giới thiệu chung:**

Cảm biến khí ga MQ2 là một trong những loại cảm biến được sử dụng để nhận biết: LPG, i-butan, Propane, Methane , Alcohol, Hydrogen, Smoke và khí gas. Trong đề tài này, với mô hình nghiên cứu, nhóm chúng tôi sử dụng Module MQ-2 để phát hiện khói.Module này được thiết kế với độ nhạy cao, thời gian đáp ứng nhanh. Giá trị đọc được từ cảm biến sẽ được đọc về từ chân Analog của vi điều khiển.

Vật liệu nhạy cảm của cảm biến khí MQ-2 là SnO2, có độ dẫn thấp trong không khí trong lành. Khi tồn tại khí đốt mục tiêu, độ dẫn điện của cảm biến sẽ cao hơn cùng với nồng độ khí tăng lên.

**Hình 2.4: Cảm biến khí MQ-2.**

* **Đặc tính:**

- Độ nhạy tốt với khí đốt trong phạm vi rộng.

- Độ nhạy cao đối với khí gas, LPG, Propane và Hydrogen.

- Tuổi thọ cao và chi phí thấp.

- Mạch đơn giản.

* **Ứng dụng:**

- Máy dò rò rỉ gas trong nước.

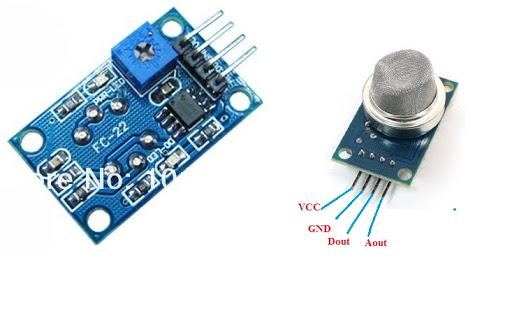
- Máy dò khí đốt công nghiệp.

- Máy dò khí xách tay.

### **2.4.2 Thông số kĩ thuật:**

* **Sơ đồ chân của Module cảm biến khí gas MQ-2:**

Module cảm biến khí gas MQ-2 là sự tích hợp thêm cho cảm biến MQ-2 một mạch điện đơn giản để khi cảm biến phát hiện khí dễ cháy thì độ nhạy của cảm biến tăng lên, cũng như độ dẫn điện cũng cao hơn dựa vào đó mạch điện đơn giản sẽ chuyển sang tín hiệu tương tự hay tín hiệu số cho người dùng tiện sử dụng.

Module được sử dụng rộng rãi trong công nghiệp và dân dụng do mạch đơn giản, nhỏ gọn và chi phí thấp. Có cả 2 tín hiệu đầu ra: Analog (A0), Digital (D0) và điện áp sử dụng 5V.

**Hình 2.5: Module cảm biến khí Gas MQ-2.**

## 2.5 Cảm biến nhiệt độ và độ ẩm của môi trường DHT11:

### **2.5.1 Giới thiệu chung:**

Ở đề tài này, với mô hình mô phỏng trong phạm vi không gian nhỏ, nhóm sinh viên sử dụng cảm biến độ ẩm và nhiệt độ DHT11 là cảm biến rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua chuẩn giao tiếp 1 wire.

- Chuẩn giao tiếp 1 wire là dùng 1 chân Digital để truyền dữ liệu.

- Bộ tiền xử lý tín hiệu được tích hợp trong cảm biến giúp bạn có thể đọc dữ liệu chính xác mà không phải qua bất kỳ tính toán nào.



**Hình 2.6: Module cảm biến nhiệt độ, độ ẩm DHT11.**

### **2.5.2 Thông số kĩ thuật:**

- Điện áp hoạt động: 3V-5V DC.

- Chuẩn giao tiếp: TTL, 1 wire.

- Dải độ ẩm hoạt động: 20%-80%RH, sai số ± 5%RH.

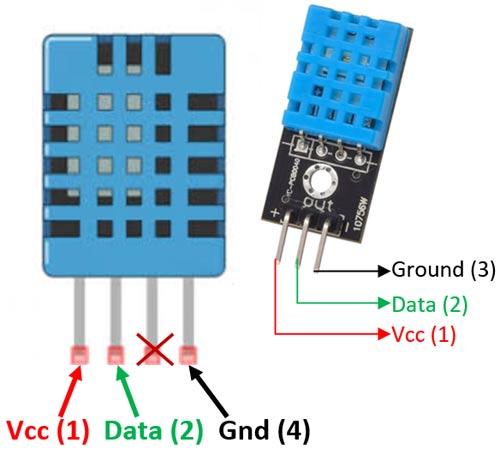
- Dải độ ẩm hoạt động: 0-50°C, sai số ± 2°C.

- Tần số lấy mẫu tối đa: 1Hz (1 giây / lần).

- Khoảng cách truyền tối đa: 20m.

- Kích thước: 28mm x 12mm x10m.

### **2.5.3 Sơ đồ chân của Module:**

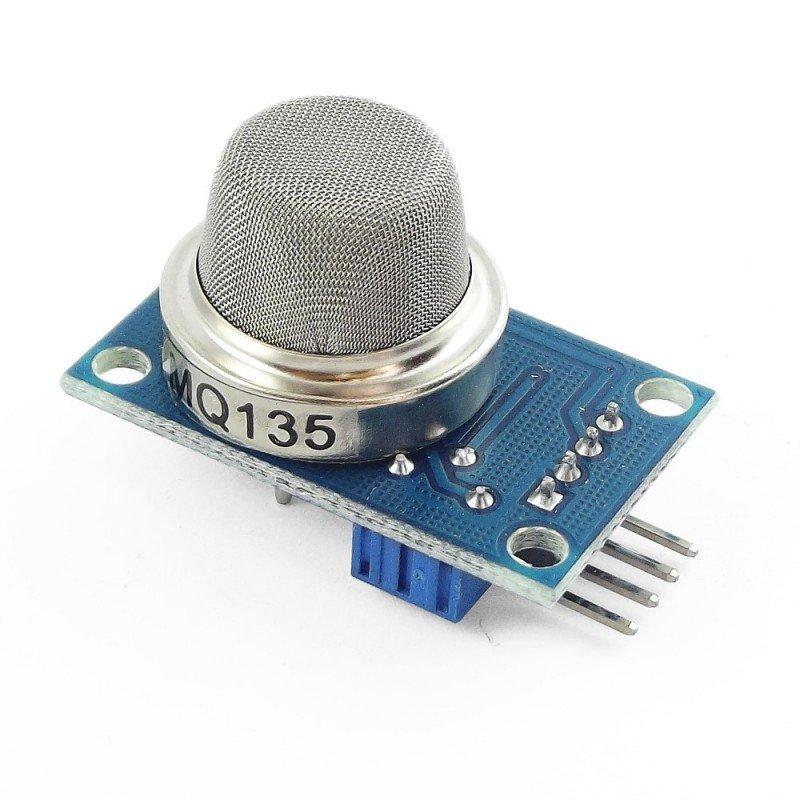


**Hình 2.7: Sơ đồ chân Module cảm biến nhiệt độ, độ ẩm DHT11.**

## 2.6 Cảm biến chất lượng không khí MQ-135:

### **2.6.1 Giới thiệu chung:**

Với chức năng phát hiện khí độc phát ra từ đám cháy hoặc từ môi trường xung quanh. Để thực hiện được điều này, nhóm sinh viên sử dụng module cảm biến chất lượng không khí MQ-135, cảm biến này có thể phát hiện được các khí như NH3, NOx, Ancol, benzen, khói, gas, CO2…. Đây đều là các tạp chất và không có lợi cho sức khỏe con người nên người ta gọi nó là cảm biến chất lượng không khí.



**Hình 2.8:Cảm biến kiểm tra chất lượng không khí MQ-135.**

### **2.6.2 Thông số kĩ thuật:**

- Điện áp nguồn: <=24V DC.

- Điện áp của heater: 5V±0.1 AC/DC.

- Điện trở tải: thay đổi được (2kΩ-47kΩ).

- Điện trở của heater: 33Ω±5%.

- Công suất tiêu thụ của heater: ít hơn 800mW.

- Nồng độ phát hiện của một số chất: 10 – 300ppm đối với NH3, 10 - 100ppm đối với bezen, 10 -300ppm đối với alcol…

- Giá trị analog trả về: 0 -1023.

Ppm là đơn vị đo mật độ thường dành cho các mật độ tương đối thấp. Nó thường chỉ tỷ lệ của một lượng chất trong hỗn hợp chứa chất đó.

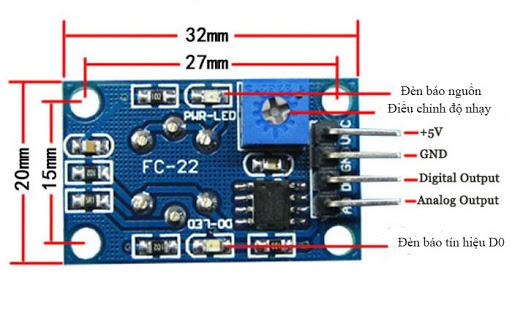
- Kích thước: 32mm\*20mm.

- Bền, tuổi thọ cao.

- Khoảng đo rộng, phát hiện nhanh, độ nhạy cao.

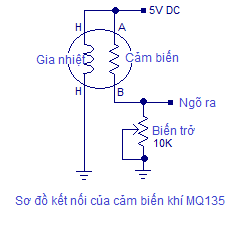
- Mạch đơn giản.

### **2.6.3 Sơ đồ bố trí và kích thước của Module MQ-135:**



**Hình 2.9: Sơ đồ bố trí và kích thước của Module MQ-135.**

* Sơ đồ kết nối của cảm biến MQ-135:



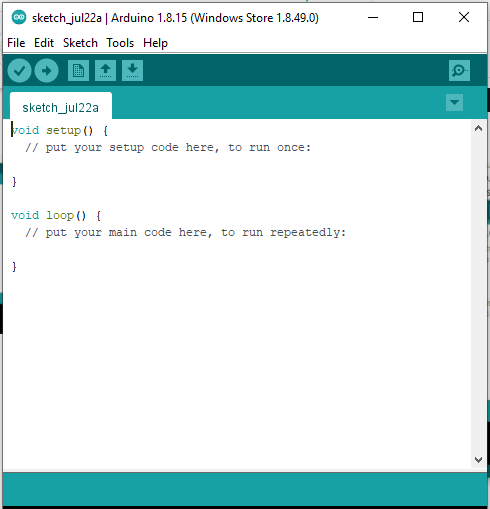
**Hình 2.10: Sơ đồ kết nối của Module MQ-135.**

## 2.7 Phần mềm:

### **2.7.1 Arduino IDE:**

Môi trường phát triển tích hợp (IDE) của Arduino là một ứng dụng cross-platform (đa nền tảng) được viết bằng Java, và từ IDE này sẽ được sử dụng cho Ngôn ngữ lập trình xử lý (Processing programming language) và project Wiring. Nó được thiết kế để dành cho những người mới tập làm quen với lĩnh vực phát triển phần mềm. Nó bao gồm một chương trình code editor với các chức năng như đánh dấu cú pháp, tự động brace matching, và tự động canh lề, cũng như compile (biên dịch) và upload chương trình lên board chỉ với 1 cú nhấp chuột. Một chương trình hoặc code viết cho Arduino được gọi là một sketch.

Các chương trình Arduino được viết bằng C hoặc C++. Arduino IDE đi kèm với một thư viện phần mềm được gọi là "Wiring", từ project Wiring gốc, có thể giúp các thao tác input/output được dễ dàng hơn. Người dùng chỉ cần định nghĩa 2 hàm để tạo ra một chương trình vòng thực thi (cyclic executive) có thể chạy được:



**Hình 2.11: Giao diện phần mềm Arduino IDE.**

• setup(): hàm này chạy mỗi khi khởi động một chương trình, dùng để thiết lập các cài đặt.

• loop(): hàm này được gọi lặp lại cho đến khi tắt nguồn board mạch.

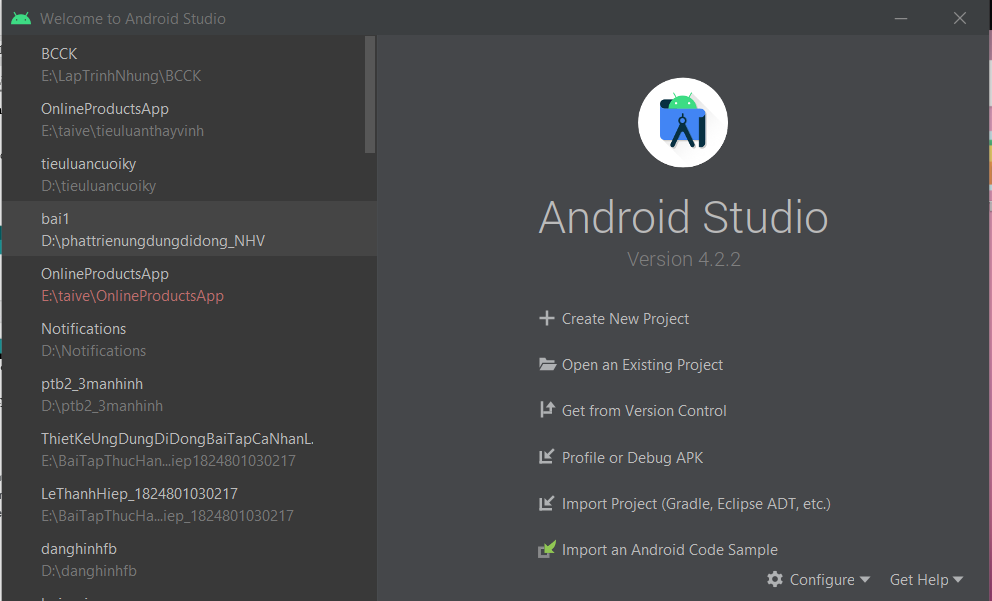
### **2.7.2 Android Studio:**

**Android Studio**là [môi trường phát triển tích hợp](https://vi.wikipedia.org/wiki/M%C3%B4i_tr%C6%B0%E1%BB%9Dng_ph%C3%A1t_tri%E1%BB%83n_t%C3%ADch_h%E1%BB%A3p) (IDE) chính thức dành cho phát triển nền tảng [Android](https://vi.wikipedia.org/wiki/Android_(h%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh)" \o "Android (hệ điều hành)).

Nó được ra mắt vào ngày 16 tháng 5 năm 2013 tại hội nghị [Google I/O](https://vi.wikipedia.org/wiki/Google_I/O" \o "Google I/O). Android Studio được phát hành miễn phí theo giấy phép [Apache Licence 2.0](https://vi.wikipedia.org/wiki/Gi%E1%BA%A5y_ph%C3%A9p_Apache" \o "Giấy phép Apache).

Android Studio ở giai đoạn truy cập xem trước sớm bắt đầu từ phiên bản 0.1 vào tháng 5.2013, sau đó bước vào giai đoạn beta từ phiên bản 0.8 được phát hành vào tháng 6 năm 2014. Phiên bản ổn định đầu tiên được ra mắt vào tháng 12 năm 2014, bắt đầu từ phiên bản 1.0.

Dựa trên phần mềm [IntelliJ IDEA](https://vi.wikipedia.org/wiki/IntelliJ_IDEA" \o "IntelliJ IDEA) của JetBrains, Android Studio được thiết kế đặc biệt để phát triển ứng dụng Android. Nó hỗ trợ các hệ điều hành [Windows](https://vi.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows), [Mac OS X](https://vi.wikipedia.org/wiki/OS_X) và [Linux](https://vi.wikipedia.org/wiki/Linux); và là IDE chính thức của Google để phát triển ứng dụng Android gốc để thay thế cho [Android Development Tools](https://vi.wikipedia.org/w/index.php?title=Android_Development_Tools&action=edit&redlink=1" \o "Android Development Tools (trang chưa được viết)) (ADT) dựa trên [Eclipse](https://vi.wikipedia.org/wiki/Eclipse_(m%C3%B4i_tr%C6%B0%E1%BB%9Dng_ph%C3%A1t_tri%E1%BB%83n_t%C3%ADch_h%E1%BB%A3p)" \o "Eclipse (môi trường phát triển tích hợp)).



**Hình 2.12: Giao diện Start khi mở phần mềm Android Studio.**

## 2.8 Chuẩn Truyền Dữ Liệu Nối Tiếp UART:

UART – là viết tắt của Universal Asynchronous Receiver – Transmitter có nghĩa là truyền nhận dữ liệu nối tiếp bất đồng bộ.

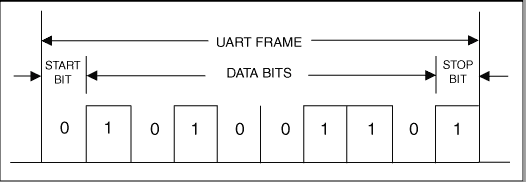
* **Đặc điểm:** Truyền dữ liệu nối tiếp bất đồng bộ có 1 đường phát dữ liệu (Tx) và 1 đường nhận dữ liệu (Rx), do không có tín hiệu xung clock nên gọi là bất đồng bộ. Để truyền được dữ liệu thì cả bên phát và bên nhận phải tự tạo xung clock có cùng tần số và thường được gọi là tốc độ baud, ví dụ như 2400 baud, 4800 baud, 9600 baud...
* **Ưu điểm:** Đơn giản, hiệu quả tương đối cao.

**- Khuyết điểm*:*** Do tồn tại các bit start và bit stop, khoảng trống dẫn đến thời gian truyền chậm.

### **2.8.1 Quá trình truyền dữ liệu UART:**

Để bắt đầu cho việc truyền dữ liệu bằng UART, một START bit được gửi đi, sau đó là các bit dữ liệu và kết thúc quá trình truyền là STOP bit.

Khi ở trạng thái chờ (idle) mức điện thế ở mức 1 (high). Khi bắt đầu truyền START bit sẽ chuyển từ 1 xuống 0 để báo hiệu cho bộ nhận là quá trình truyền dữ liệu sắp xảy ra. Sau START bit là đến các bit dữ liệu D0 - D7 (các bit này có thể ở mức High hoặc Low tùy theo dữ liệu, theo hình ví dụ như trên byte dữ liệu là LSB – 11010010 – MSB). Sau khi truyền hết dữ liệu thì đến bit kiểm tra Parity. Cuối cùng là STOP bit là 1 báo cho thiết bị rằng các bit đã được gửi xong. Thiết bị nhận sẽ tiến hành kiểm tra khung truyền nhằm đảm báo tính đúng đắn của dữ liệu.



**Hình 2.13: Cấu trúc cơ bản của 1 frame dữ liệu.**

Để quá trình truyền thành công thì việc tuân thủ các tiêu chuẩn truyền là hết sức quan trọng. Vì vậy cần phải nắm rõ các khái niệm quan trọng trong phương thức truyền này, sẽ được trình bày ở nội dung tiếp theo.

### **2.8.2 Thông số chuẩn truyền UART:**

* ***Tốc độ truyền (baud rate):***

Như trong ví dụ trên về việc truyền 1 bit trong 1ms, để việc truyền và nhận bất đồng bộ xảy ra thành công thì các thiết bị tham gia phải thống nhất nhau về khoảng thời gian dành cho 1 bit truyền, hay nói cách khác tốc độ truyền phải được cài đặt như nhau trước, tốc độ này gọi là tốc độ baud. Theo định nghĩa, tốc độ baud là số bit truyền trong 1 giây. Ví dụ nếu tốc độ baud được đặt là 19200 thì thời gian dành cho 1 bit truyền là 1/19200 ~ 52.083us.3

* ***Khung dữ liệu (frame):***

Do truyền thông nối tiếp và nhất là nối tiếp bất đồng bộ nên rất dễ mất hoặc sai lệch dữ liệu, quá trình truyền theo kiểu này phải tuân theo một số quy cách nhất định. Bên cạnh tốc độ baud, khung truyền là một yếu tố quan trọng tạo nên sự thành công khi truyền và nhận. Khung truyền bao gồm các quy định về số bit trong mỗi lần truyền, các bit “báo” như bit Start và bit Stop, các bit kiểm tra như Parity, ngoài ra số lượng các bit trong một data cũng được quy định bởi khung truyền.

* ***Start bit:***

Start là bit đầu tiên được truyền trong một frame truyền, bit này có chức năng báo cho thiết bị nhận biết rằng có một gói dữ liệu sắp được truyền tới. Start bit là bit bắt buộc phải có trong khung truyền. Đối với chuẩn truyền UART Start bit luôn luôn là mức thấp (0V).

* ***Data:***

Data hay dữ liệu cần truyền là thông tin chính mà chúng ta cần gởi và nhận. Data không nhất thiết phải là gói 8 bit. Trong truyền nối tiếp UART, bit có ảnh hưởng nhỏ nhất (Least Significant Bit, bit bên phải) của data sẽ được truyền trước và cuối cùng là bit có ảnh hưởng lớn nhất (Most Significant Bit, bit bên trái).

* ***Parity bit:***

Parity là bit dùng kiểm tra tính đúng đắn của dữ liệu truyền (một cách tương đối). Có 2 loại parity là parity chẵn (even parity) và parity lẻ (odd parity). Parity chẵn nghĩa là số lượng bit 1 trong dữ liệu bao gồm bit parity luôn là số chẵn. Ngược lại tổng số lượng các bit 1 nếu là parity lẻ thì luôn là số lẻ. Ví dụ, nếu dữ liệu là 10111011 nhị phân, có tất cả 6 bit 1 trong dữ liệu này, nếu parity chẵn được dùng, bit parity sẽ mang giá trị 0 để đảm bảo tổng số các bit 1 là số chẵn (6 bit 1).

* ***Stop bit:***

Stop bits là một hoặc các bit báo cho thiết bị nhận biết rằng một gói dữ liệu đã được gởi xong. Sau khi nhận được stop bits, thiết bị nhận sẽ tiến hành kiểm tra khung truyền để đảm bảo tính chính xác của dữ liệu. Stop bits là các bits bắt buộc xuất hiện trong khung truyền và luôn luôn là mức cao.

## 2.9 Sử dụng Website firebase.google.com:

### **2.9.1 Giới thiệu:**

Firebase là một nền tảng để phát triển ứng dụng di động và trang web, bao gồm các API đơn giản và mạnh mẽ mà không cần backend hay server.

Firebase là dịch vụ cơ sở dữ liệu hoạt động trên nền tảng đám mây – cloud. Kèm theo đó là hệ thống máy chủ cực kỳ mạnh mẽ của Google. Chức năng chính là giúp người dùng lập trình ứng dụng bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu. Cụ thể là những giao diện lập trình ứng dụng API đơn giản.

Đặc biệt, còn là dịch vụ đa năng và bảo mật cực tốt. Firebase hỗ trợ cả hai nền tảng Android và IOS. Không có gì khó hiểu khi nhiều lập trình viên chọn Firebase làm nền tảng đầu tiên để xây dựng ứng dụng cho hàng triệu người dùng trên toàn thế giới.

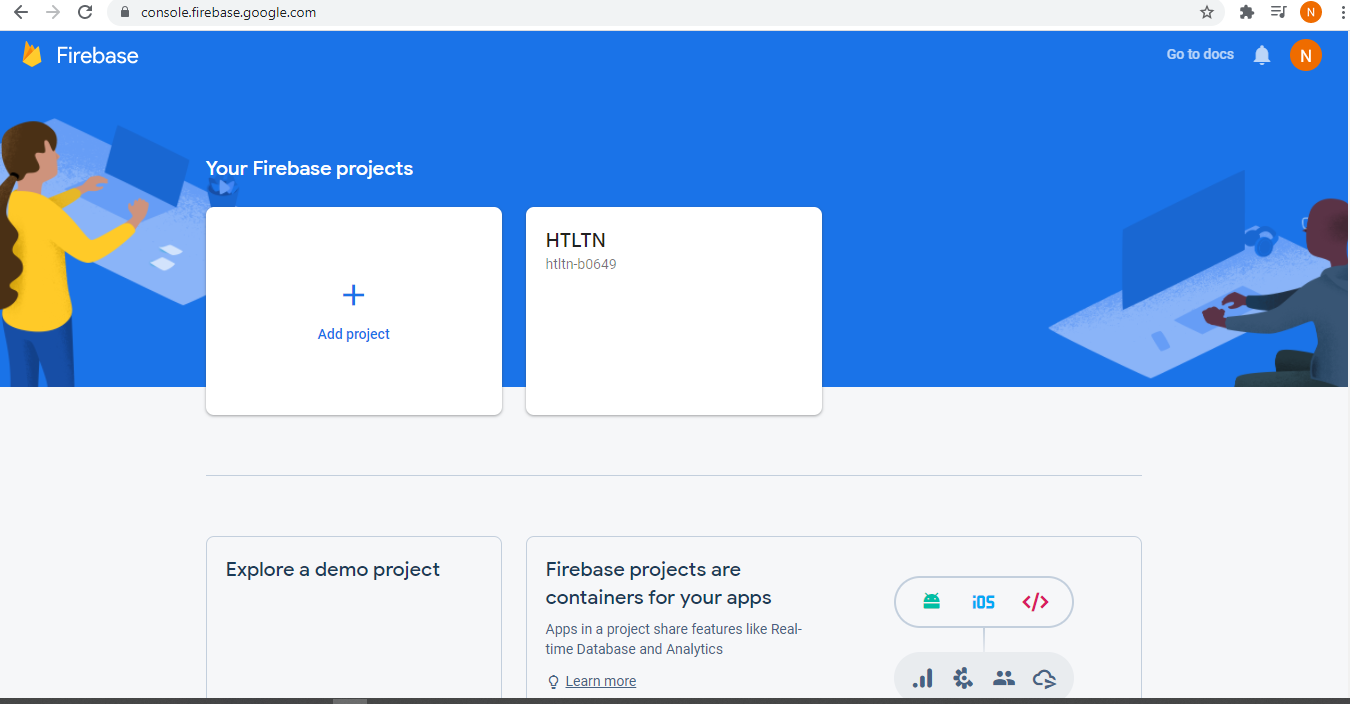
* ***Cách thức hoạt động của Firebase bao gồm*** *:*
* Firebase Realtime Database: Khi đăng ký một tài khoản trên Firebase để tạo ứng dụng, bạn đã có một cơ sở dữ liệu thời gian thực. Dữ liệu bạn nhận được dưới dạng JSON. Đồng thời nó cũng luôn được đồng bộ thời gian thực đến mọi kết nối client.
* Firebase Authentication: Firebase xây dựng các bước xác thực người dùng bằng Email, Facebook, Twitter, GitHub, Google. Đồng thời cũng xác thực nặc danh cho các ứng dụng. Hoạt động xác thực có thể giúp thông tin cá nhân của người sử dụng được an toàn và đảm bảo không bị đánh cắp tài khoản.
* Firebase Hosting: Firebase cung cấp các hosting. Hosting được phân phối qua tiêu chuẩn công nghệ bảo mật SSL từ mạng CDN (Content Delivery Network là mạng lưới máy chủ lưu giữ bản sao của các nội dung tĩnh bên trong website và phân phối đến nhiều máy chủ PoP). Hoạt động này giúp lập trình viên tiết kiệm thời gian thiết kế, xây dựng và phát triển ứng dụng.
* **Ưu điểm:**
* Nhiều dịch vụ trong một nền tảng.
* Tập trung vào phát triển giao diện người dùng.
* Firebase không có máy chủ.
* Theo dõi lỗi
* Sao lưu.
* **Nhược điểm:**
* Không phải là mã nguồn mở, người dùng không có quyền truy cập mã nguồn.

- Firebase không hoạt động ở nhiều quốc gia.

### **2.9.2 Sử dụng nền tảng Firebase:**

* **Để sử dụng nền tảng Firebase ta cần có chuẩn bị các bước sau:**
* **Bước 1:** Đăng nhập tài khoản firebase.google.com:

Bạn truy cập vào địa chỉ https:// firebase.google.com và chọn Sign In để đăng nhập tài khoản. Nền tàng được phát triển bởi Google nên bạn có thể sử dụng hoàn toàn miễn phí với tài khoản Google của mình.

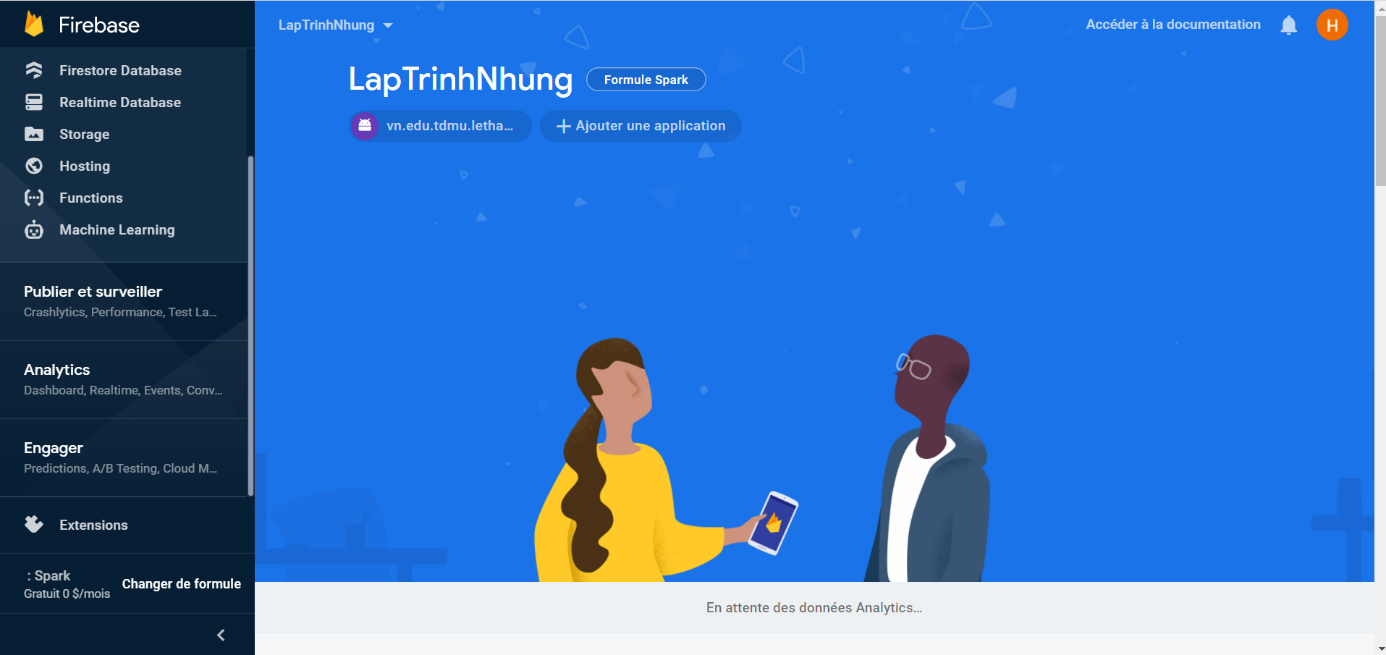


**Hình 2.14: Trang chủ Firebase sau khi Sign In thành công.**

* **Bước 2:** Tạo Project:

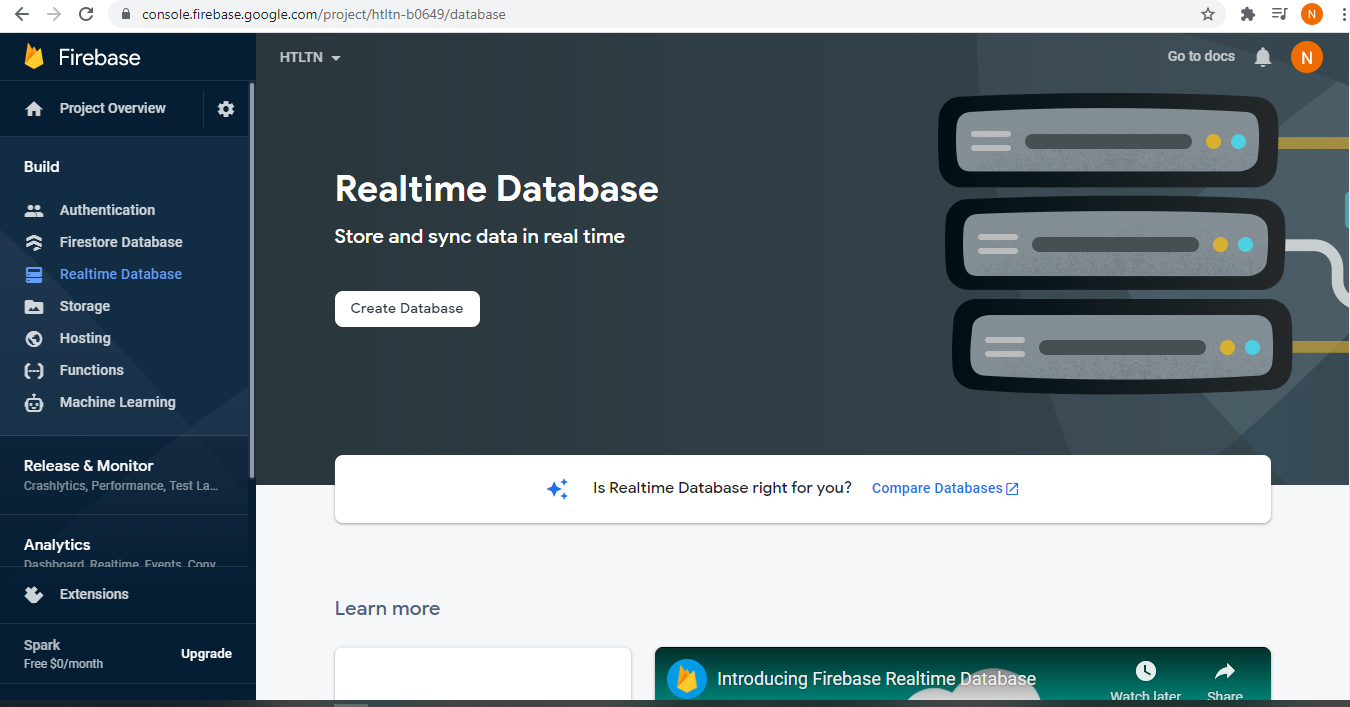
Sau khi login, bạn cần tạo một Project để tạo môi trường lưu trữ dữ liệu. Trên trang, bạn chọn Add Project và điền các thông tin cần thiết.

Ở đây mình tạo 1 Project với tên “LapTrinhNhung” để chứa các dữ liệu từ cảm biến thu được từ thiết bị của mình.



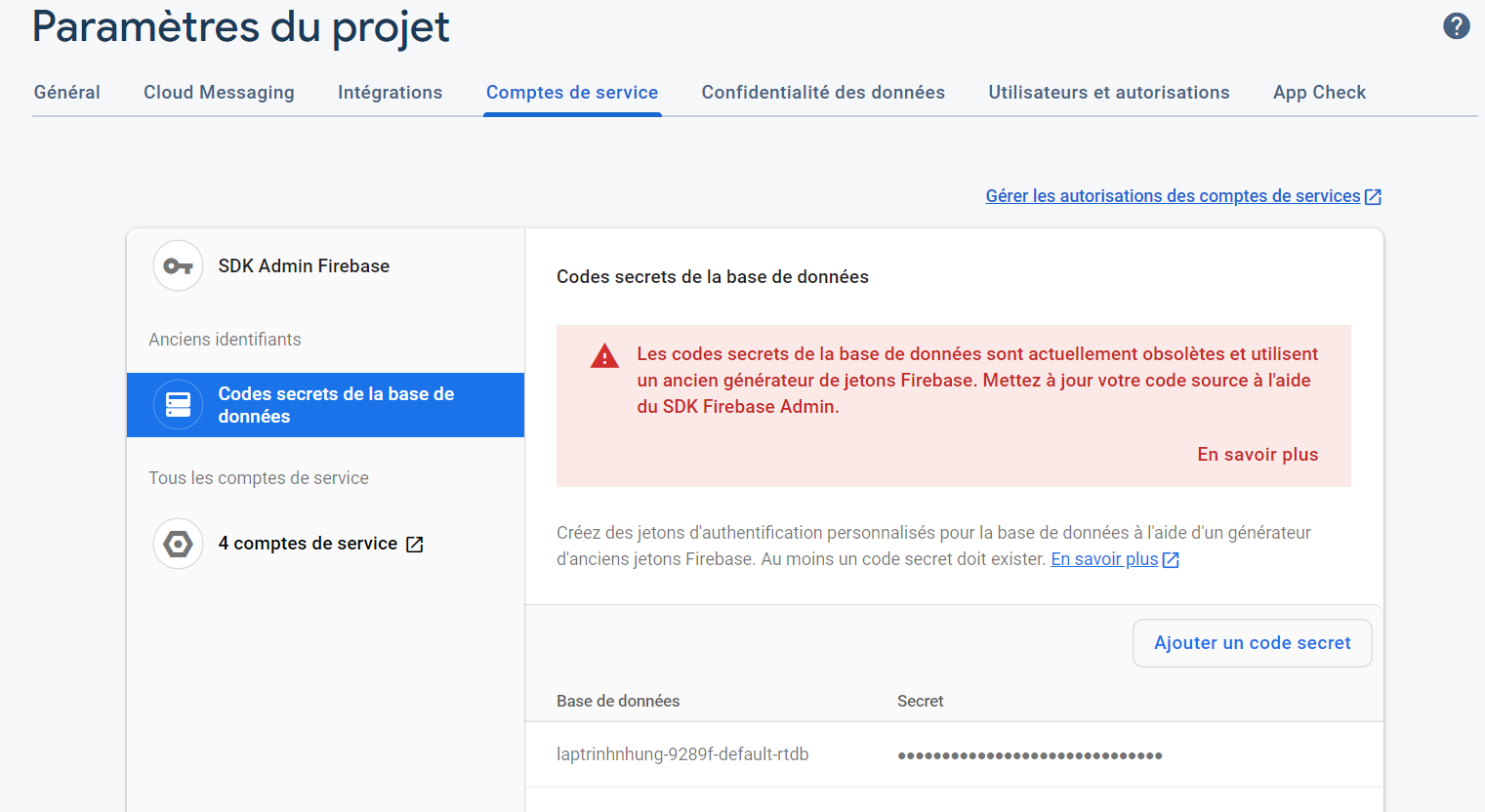
**Hình 2.15: Project đã Add thành công.**

Bạn chọn sử dụng công cụ Realtime Database. Sau đó nhấn Create Database:

****

**Hình 2.16: Giao diện Trang Firebase Realtime Database.**

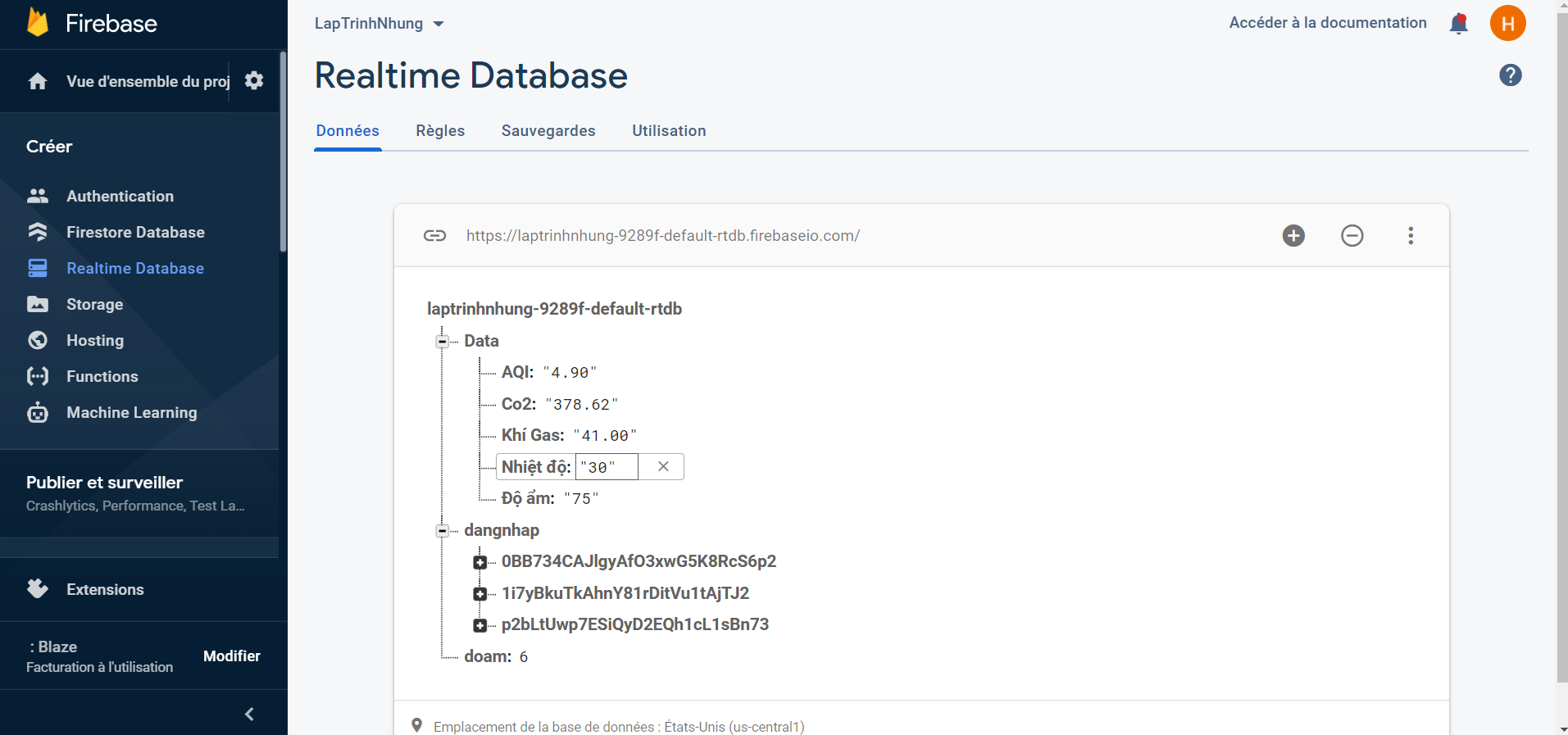
* **Bước 3:** Lấy mã CSDL trong setting:



**Hình 2.17: Lấy mã để truyền data.**

* **Bước 4:** Upload dữ liệu từ ESP8266 NodeMCU:

Sau khi ESP8266 NodeMCU chạy và bắt đầu gữi dữ liệu lên Firebase cloud, bạn có thể xem các dữ liệu này trong tại firebase realtime database



**Hình 2.18: Dữ liệu được upload lên.**

# CHƯƠNG III: THIẾT KẾ HỆ THỐNG

## 3.1 Yêu cầu của hệ thống:

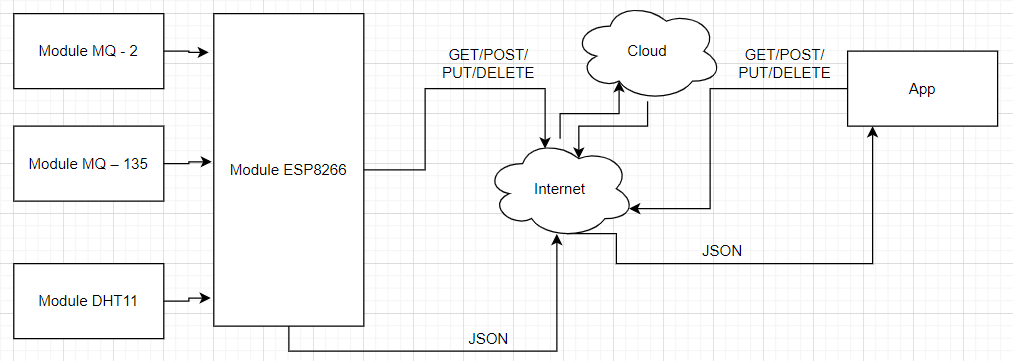
- Phần cứng (các linh kiện điện tử): Vi điều khiển, module Wi-Fi, cảm biến, dây cáp (cấp nguồn và nạp code),.. đảm bảo hoạt động tốt để các thông số của môi trường có độ chính xác cao.

- Phần mềm (App): phải đảm bảo hoạt động ổn định để cập nhật liên tục giá trị đo được của các cảm biến thông qua kết nối Internet. Đảm bảo thông báo chính xác và đồng bộ theo thời gian thực để người sử dụng có thể giám sát một cách kịp thời và hiệu quả.

## 3.2 Thiết kế phần cứng:

### **3.2.1 Sơ đồ khối hệ thống và chức năng:**

* **Sơ đồ khối:**

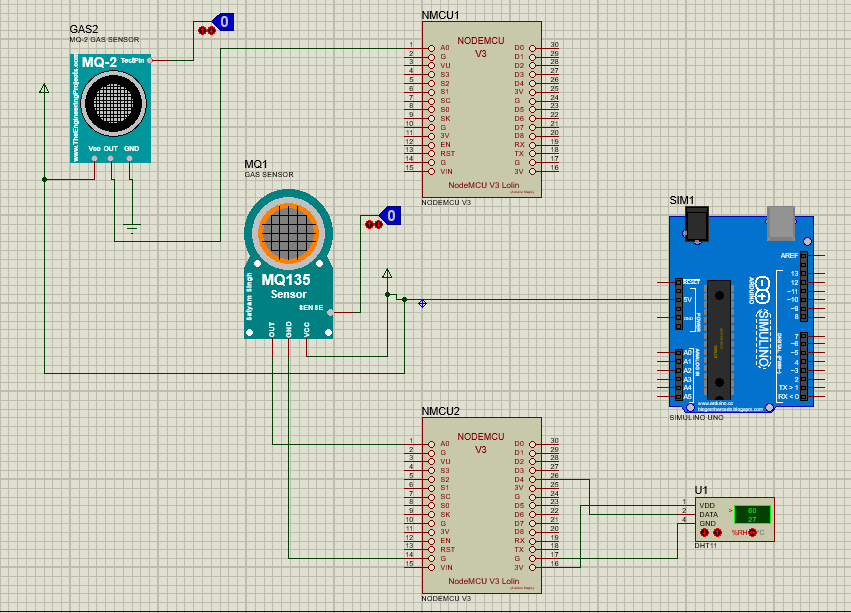


**Hình 3.1: Sơ đồ khối hệ thống.**

* **Chức năng các khối:**
* ***Khối cảm biến hỏa hoạn và khí độc (MQ-2, MQ-135, DHT11):*** Đo nồng độ khí độc trong không khí, nhiệt độ và độ ẩm của môi trường. Và gửi dữ liệu về Module ESP8266 NodeMCU xử lí.
* ***Khối xử lý trung tâm (Kit ESP8266 NodeMCU):*** thu thập dữ liệu giá trị khí gas, nhiệt độ và độ ẩm, tình trạng hoạt động của thiết bị lên App di động, hiển thị thông báo trạng thái an toàn hay nguy hiểm; đồng thời gửi dữ liệu lên Website (Cloud) để quản lý dữ liệu lâu dài thông qua kết nối Internet (sóng Wifi).

### **3.2.2 Sơ đồ nguyên lý hệ thống:**

* **Sơ đồ nguyên lý:**

****

**Hình 3.2: Sơ đồ nguyên lý hệ thống.**

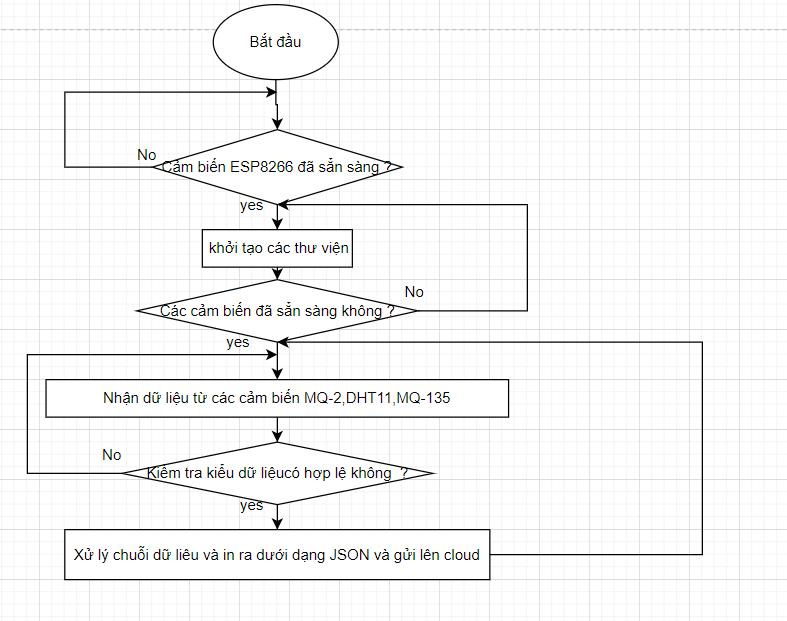
* **Bảng kết kết nối các linh kiện điện tử:**

|  |  |
| --- | --- |
| **ARDUINO** | **MQ-2** |
| 5V | VCC |
| **ARDUINO** | **MQ-135** |
| 5V | VCC |
| **ESP8266 NodeMCU** | **MQ-2** |
| GND | GND |
| A0 | A0 |
| **ESP8266 NodeMCU** | **MQ-135** |
| A0 | A0 |
| GND | GND |
| **ESP8266 NodeMCU** | **DHT-11** |
| 3V | VDD |
| GND | GND |
| D4 | DATA |

**Bảng 3.1: Bảng kết nôi dây các linh kiện điện tử.**

### **3.2.3 Lưu đồ giải thuật:**

* **Lưu đồ giải thuật tại phần cứng:**



**Hình 3.3: Sơ đồ giải thuật phần cứng.**

* ***Giải thích lưu đồ phần cứng:***

Sau khi được cấp một nguồn điện nhất định cho các model ,sau đó model ESP8266 kết nối Internet nếu kết nối thất bại thì sẽ kết nối lại còn nếu kết nối thành công thì khởi tạo các thư viện DHT11, MQ-135, MQ-2 và các cảm biến đã sẳn sàng không nếu chưa thì khởi tạo tiếp thư viện còn nếu đã sẵn sàng thì nhận dữ liệu từ các model trên và kiểm tra kiểu dữ liệu nếu kiểu dữ liệu không hợp lệ thì nhận lại dự liệu còn nếu đã sẵn sàng thì chuyển đổi dữ liệu đó thành dạng JSON và sau đó gửi lên Cloud thông qua giao thức HTPPS.

### **3.2.4 Tính toán các khối:**

* **Tính toán khối cảm biến:**
* Cảm biến đo đạc giá trị của khí gas rò rỉ bên ngoài môi trường:

+ Điện áp hoạt động của module cảm biến khí gas MQ-2 là 5V.

+ Dòng tiêu thụ là 180mA.

* Cảm biến đo đạc giá trị chất lượng của không khí :

+ Điện áp hoạt động của module cảm biến khí gas MQ-135 là 5V.

+ Dòng tiêu thụ là 150mA.

* Cảm biến đo đạc giá trị nhiệt độ, độ ẩm của môi trường:

+ Điện áp hoạt động của module cảm biến nhiệt độ, độ ẩm DHT11 là 5V.

+ Dòng tiêu thụ là 2.5mA.

* **Tính toán khối đưa dữ liệu lên Web:**
* Khối đưa dữ liệu lên Web dùng Kit ESP8266 NodeMCU. Điện áp hoạt động của ESP8266 NodeMCU là 5V và dòng hoạt động là 300mA.
* **Tính toán điện áp và dòng điện cho khối nguồn:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên linh kiện** | **Dòng định mức (mA)** | **Số lượng** | **Tổng dòng điện(mA)** |
| **1** | Cảm biến MQ-2 | 180 | 1 | 180ca |
| **2** | Cảm biến MQ-135 | 150 | 1 | 150 |
| **3** | Cảm biến DHT11 | 2.5 | 1 | 2.5 |
| **4** | Module ESP8266 NodeMCU | 300 | 2 | 600 |
| **Tổng cộng** | | | | 932.5 |

**Bảng 3.2: Bảng thống kê điện năng tiêu thụ của các linh kiện.**

* **Nguyên lý hoạt động của toàn hệ thống:**

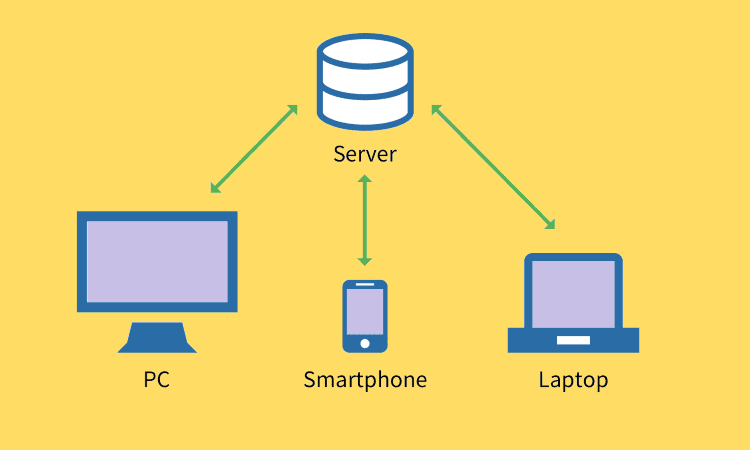
Khi cấm nguồn 5V vào cổng của Arduino, Arduino từ đó cung cấp nguồn vào sau đó điều chỉnh xuất ra mức điện áp 5V-2A ổn định để nuôi toàn bộ hệ thống. Cảm biến khí gas MQ-2, cảm biến chất lượng không khí MQ-135 và cảm biến nhiệt độ, độ ẩm hoạt động liên tục cập nhật dữ liệu gửi đến bộ xử lí (ESP8266 NodeMCU). Giá trị nhận được từ cảm biến sẽ được vi điều khiển xuất lên màn hình Smartphone qua giao diện App đã được cài đặt sẵn, đồng thời gửi dữ liệu lên Website Firebase thông qua Module ESP8266 NodeMCU để lưu trữ tại đây. Trên màn hình Smartphone báo thiết bị đang hoạt động và thông báo tình trạng lượng khí gas, khí độc tồn tại trong môi trường có an toàn, nhiệt độ và độ ẩm của môi trường có ổn định ở mức bình thường hay không.

Nếu trong trường hợp nồng độ khí gas và khí độc chưa vượt mức cho phép, nhiệt độ và độ ẩm của môi trường ổn định ở mức bình thường thì giao diện cảnh báo không có tín hiệu và hiển thị. Thanh mức giao diện màu xanh trên thiết bị Smartphone báo đang hoạt động bình thường.

Ngược lại, trong trường hợp nồng độ khí gas và khí độc vượt mức an toàn, nhiệt độ và độ ẩm của môi trường không ổn định thì ngay lập tức thông báo nồng độ khí gas (độc) bị rò rỉ, nhiệt độ và độ ẩm vượt mức bình thường trên màn hình Smartphone. Thanh mức giao diện màu đỏ trên thiết bị Smartphone cảnh báo môi trường nguy hiểm.

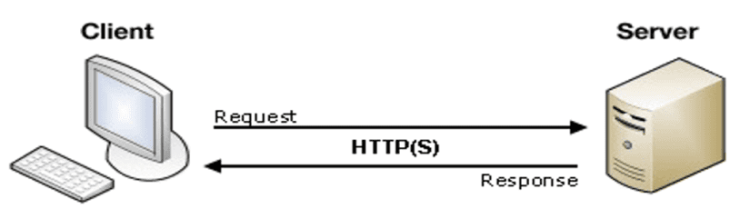
## 3.3 Thiết kế phần mềm:

### **3.3.1 Thiết kế kiến trúc:**



**Hình 3.4: Mô hình kiến trúc phần mềm.**

* **Mô hình được thiết kế theo mô hình Client – Server:**
* ***Khái niệm:*** **Mô hình Client Server** là mô hình mạng máy tính trong đó các máy tính con được đóng vai trò như một máy khách, chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ. Để máy chủ xử lý yêu cầu và trả kết quả về cho máy khách đó.
* ***Nguyên tắc hoạt động:*** Trong mô hình Client Server, server chấp nhận tất cả các yêu cầu hợp lệ từ mọi nơi khác nhau trên Internet, sau đó trả kết quả về máy tính đã gửi yêu cầu đó .Máy tính được coi là máy khách khi chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ và đợi câu trả lời được gửi về.



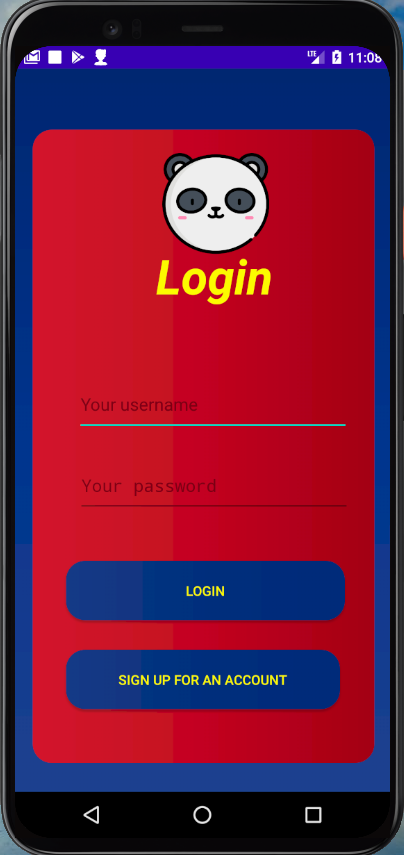
**Hình 3.5: Mô hình Client – Server.**

* **Ưu điểm của mô hình**:
* Giúp chúng ta có thể làm việc trên bất kì một máy tính nào có hỗ trợ giao thức truyền thông. Giao thức chuẩn này cũng giúp các nhà sản xuất tích hợp lên nhiều sản phẩm khác nhau mà không gặp phải khó khăn gì.
* Có thể có nhiều server cùng làm một dịch vụ, chúng có thể nằm trên nhiều máy tính hoặc một máy tính.
* Chỉ mang đặc điểm của phần mềm mà không hề liên quan đến phần cứng, ngoài yêu cầu duy nhất là server phải có cấu hình cao hơn các client.
* Hỗ trợ người dùng nhiều dịch vụ đa dạng và sự tiện dụng bởi khả năng truy cập từ xa.
* Cung cấp một nền tảng lý tưởng, cho phép cung cấp tích hợp các kỹ thuật hiện đại như mô hình thiết kế hướng đối tượng, hệ chuyên gia, hệ thông tin địa lý (GIS).
* **Nhược điểm:**

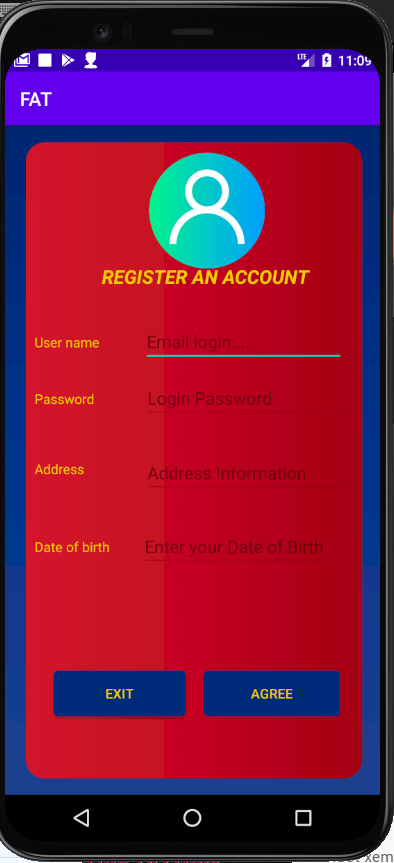
- Vấn đề bảo mật dữ liệu thông tin đôi khi còn chưa được an toàn lắm. Vì do phải trao đổi dữ liệu giữa 2 máy tính khác nhau ở 2 khu vực địa lý cách xa nhau. Và đây cũng nhược điểm duy nhất của mô hình này.

- Tuy nhiên vấn đề này thì có một số giao thức đã hỗ trợ bảo mật dữ liệu khi truyền tải. Giao thức được sử dụng phổ biến như HTTPS.

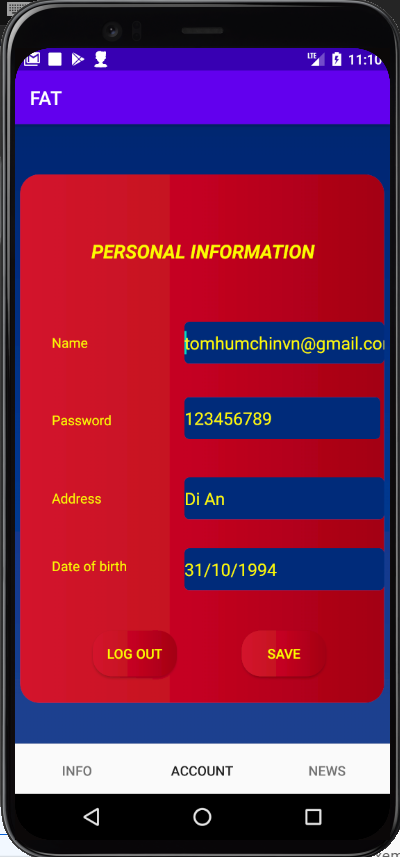
### **3.3.2 Thiết kế giao diện:**



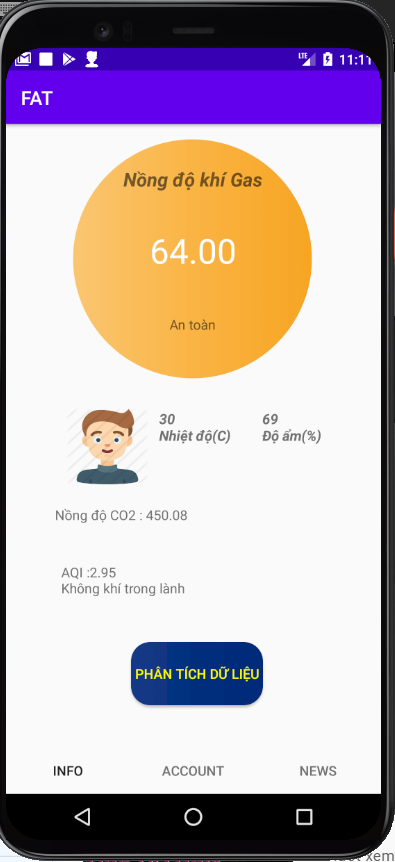
**Hình 3.6: Giao diện Đăng nhập tài khoản để sử dụng phần mềm.**



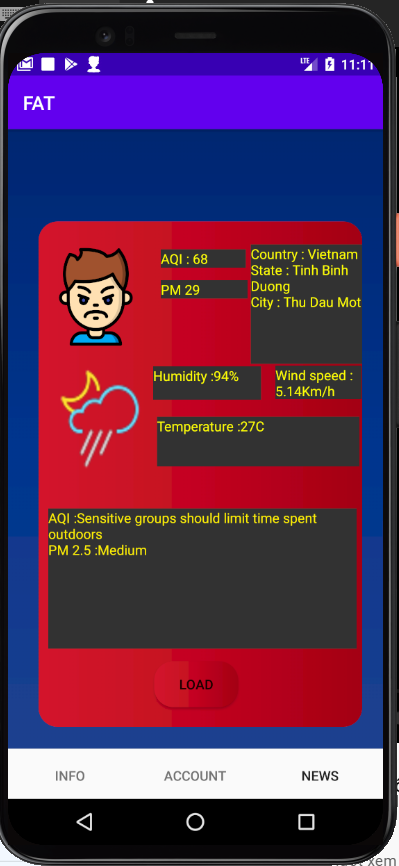
**Hình 3.7: Giao diện Đăng kí tài khoản để sử dụng phần mềm.**



**Hình 3.8: Giao diện Thông tin tài khoản cá nhân.**

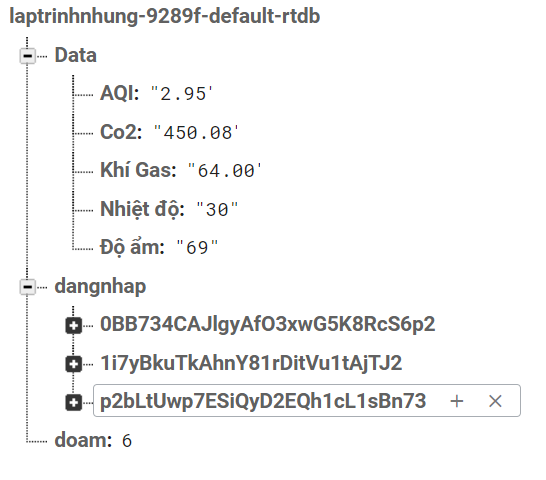


**Hình 3.9: Giao diện Quản lý hệ thống.**



**Hình 3.10: Giao diện Thông báo giá trị các thông số của cảm biến.**

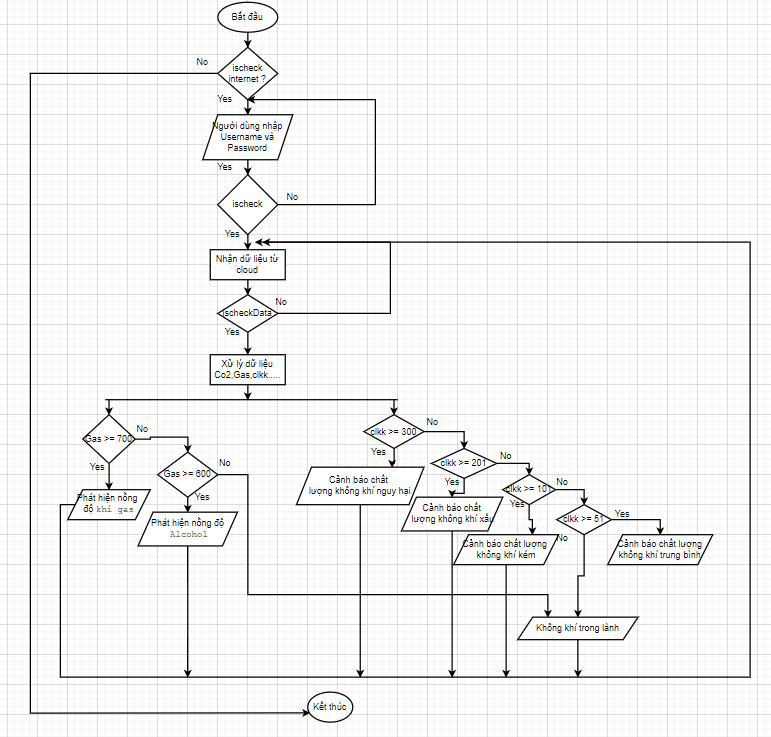
### **3.3.3 Thiết kế dữ liệu:**



**Hình 3.11: Cấu trúc dữ liệu.**

### **3.3.4 Thiết kế thuật toán:**

* **Lưu đồ giải thuật phần mềm:**



**Hình 3.12: Lưu đồ giải thuật phần mềm.**

* ***Giải thích:***

Khi vào App sẽ có hàm kiểm tra internet nếu không có thì người dùng không thể đăng nhập vào hệ thống còn nếu có thì người dùng nhập mật khẩu nếu đúng thì sẽ được truy cập hệ thống, sau khi người dùng đăng nhập thành công và truy cập vào mục INFO app sẽ nhận dữ liệu từ Cloud và xử lý chúng.

App sẽ thông báo có khí Gas khi chỉ số khí lớn hơn hoặc bằng 700 thì sẽ thông báo có nồng độ khí Gas.

App sẽ thông báo có khí Gas khi chỉ số khí lớn hơn hoặc bằng 600 thì sẽ thông báo có nồng độ khí Alcohol.

App sẽ thông báo AQI khi chỉ số khí lớn hơn hoặc bằng 300 thì sẽ thông báo chất lượng không khí nguy hại.

App sẽ thông báo AQI khi chỉ số khí lớn hơn hoặc bằng 201 thì sẽ thông báo chất lượng không khí xấu.

App sẽ thông báo AQI khi chỉ số khí lớn hơn hoặc bằng 101 thì sẽ thông báo chất lượng không khí kém.

App sẽ thông báo AQI khi chỉ số khí lớn hơn hoặc bằng 51 thì sẽ thông báo chất lượng không khí trung bình.

Còn trường hợp khác thì app sẽ thông báo không khí trong lành.

## 3.4 Hướng dẫn sử dụng và thao tác:

### **3.4.1 Hướng dẫn sử dụng:**

***- Hệ thống cảnh báo hỏa hoạn và khí độc được thiết kế gồm ba phần:***

+ Trạm xử lý.

+ Trung tâm xử lý.

+ Giao diện giám sát.

Hệ thống được thiết kế gồm ba phần nhằm đơn giản hóa thao tác vận hành cho người sử dụng.

Đối với trạm xử lý và trung tâm xử lý: để chúng hoạt động, người sử dụng chỉ cần cấp nguồn cho hệ thống thông qua Adapter.

Đối với giao diện giám sát: ngoài việc cấp nguồn thì người sử dụng cần phải kết nối giao diện giám sát và trung tâm xử lý lại với nhau thông qua kết nối Internet (sóng Wifi).

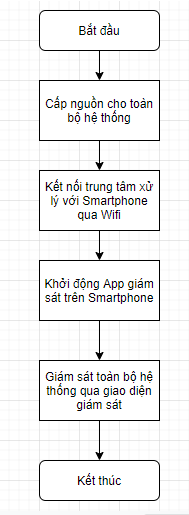
* ***Các bước thao tác dưới đây sẽ giúp chúng ta dễ hình dung hơn về việc vận hành và sử dụng hệ thống:***

+ Bước 1: Cấp nguồn cho toàn bộ hệ thống thông qua các Adapter.

+ Bước 2: Kết nối trung tâm xử lý với điện thoại qua cùng Wifi và khởi động giao diện giám sát trên điện thoại.

+ Bước 3: Giám sát toàn bộ hệ thống thông qua giao diện giám sát.

### **3.4.2 Quy trình thao tác:**



**Hình 3.13: Quy trình thao tác hệ thống cảnh báo hoả hoạn và khí độc.**

Bắt đầu chương trình, hệ thống tiến hành kiểm tra kết nối giữa giao diện giám sát và trung tâm xử lý, nếu đã được kết nối thì cho phép giao diện hoạt động. Lúc này giao diện sẽ nhận và hiển thị thông tin được gửi lên từ trung tâm xử lý. Nếu có tín hiệu báo động từ trung tâm xử lý thì giao diện sẽ thay đổi trạng thái giao diện để thông báo cho người giám sát. Sau cùng, dữ liệu được gửi lên từ trung tâm xử lý sẽ được lưu trữ lại trên Firebase thông qua Wifi.

# CHƯƠNG IV: KẾT QUẢ THỰC HIỆN

## 4.1 Phần cứng:

### **4.1.1 Thi công boarb mạch hệ thống:**

* ***Linh kiện cần chuẩn bị:***

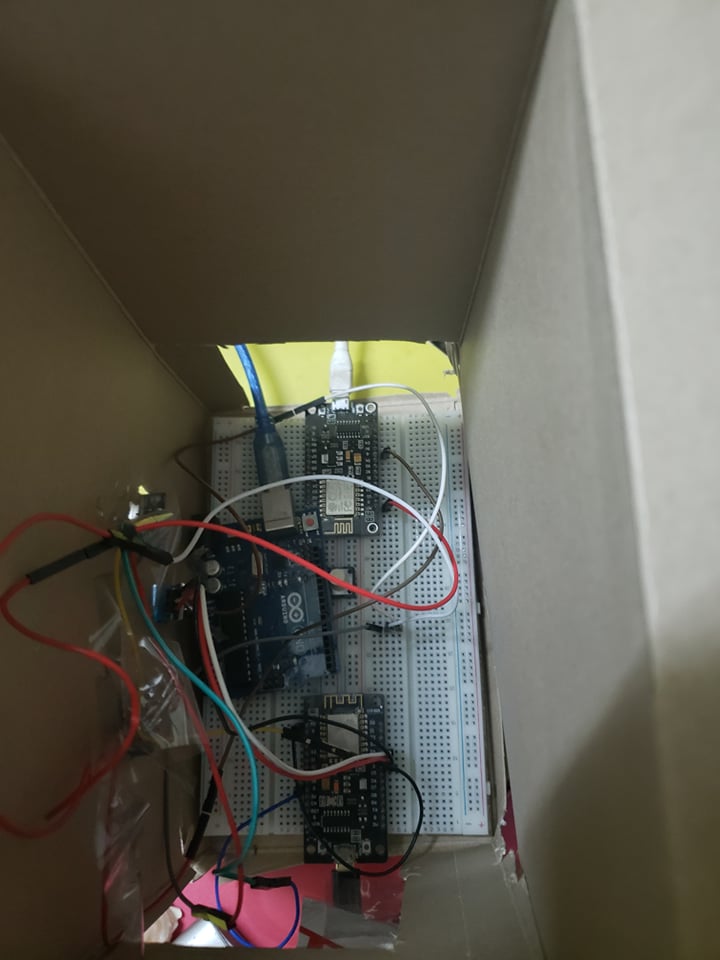
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên linh kiện** | **Số lượng** | **Đơn giá** | **Thành tiền** |
| 1 | Cảm biến khí gas MQ-2 | 1 | 32.000 | 32.000 |
| 2 | Cảm biến không khí MQ-135 | 1 | 38.000 | 38.000 |
| 3 | Cảm biến nhiệt độ, độ ẩm DHT11 | 1 | 37.000 | 37.000 |
| 4 | Module ESP8266 NodeMCU | 2 | 62.000 | 124.000 |
| 5 | Breadboard | 2 | 20.000 | 40.000 |
| Tổng cộng | | | | 271.000 |

**Bảng 4.1: Bảng linh kiện cần chuẩn bị.**

* ***Lắp ráp và kiểm tra mạch:***

Sau khi chuẩn bị đầy đủ linh kiện ta tiến hành hàn các linh kiện lên mạch thông qua các chân cắm để dễ dàng thay thế và hiệu chỉnh trong trường hợp cần thiết.

Lắp ráp các module, linh kiện lên mạch cho hoàn chỉnh. Sau khi hoàn tất, tiến hành kiểm tra kết nối có tốt không, nếu trong trường hợp xảy ra lỗi cần tiến hành khắc phục ngay sau đó kiểm tra lại.



**Hình 4.1: Cắm và kết nối các linh kiện lên Breadboard.**

### **4.1.2 Thi công hộp bảo vệ:**

* ***Chuẩn bị vật liệu:***
* Băng keo trong suốt:



**Hình 4.2: Băng keo trong.**

* Dao cắt giấy, lưỡi dao:

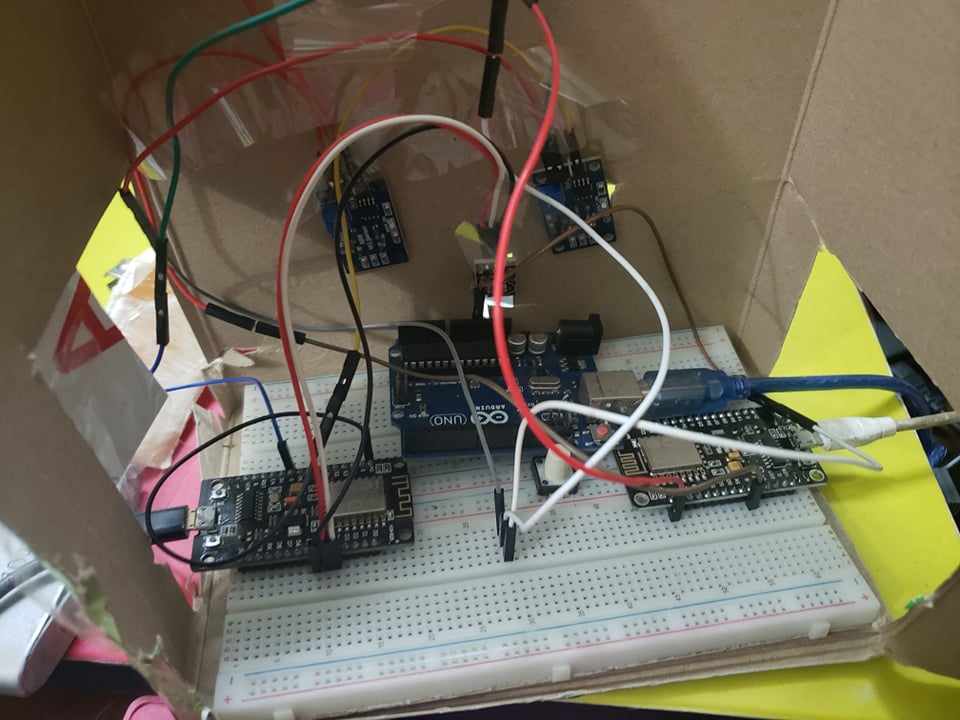
**Hình 4.3: Dao cắt giấy.**

* Hộp bảo vệ : chất liệu giấy bìa cứng.

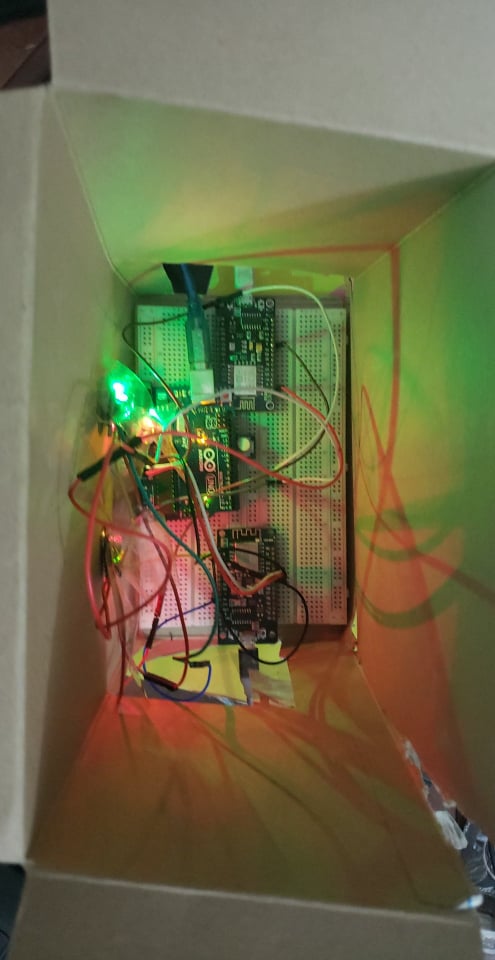


**Hình 4.4: Hộp bảo vệ.**

* ***Hộp bảo vệ sau khi hoàn thành:***

****

**Hình 4.5: Hệ thống chưa mở.**

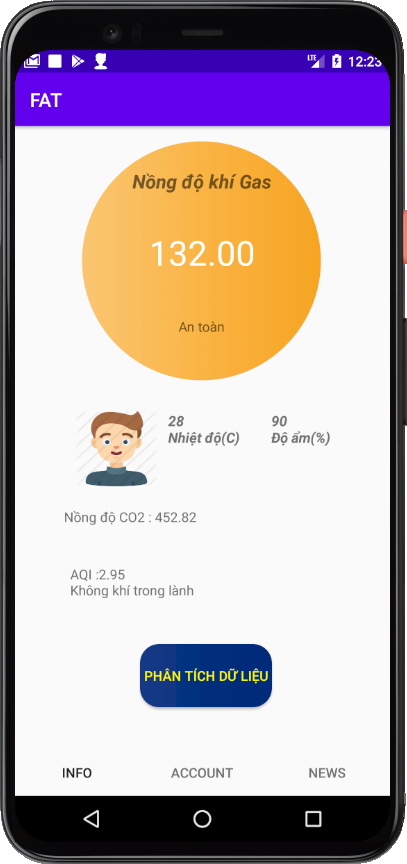


**Hình 4.6: Hệ thống đang hoạt động.**

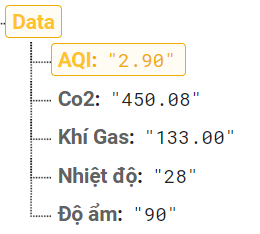
## 4.2 Giao diện phần mềm:

### **4.2.1 Hệ thống đang hoạt động không có sự cố rò rỉ khí gas:**

Khi không có khí gas bị rò rỉ cảm biến liên tục cập nhật giá trị và so sánh với mức nguy hiểm và hiển thị giá trị lên màn hình. Đồng thời hiển thị thông báo không có khí gas bị rò và khu vực an toàn. Dữ liệu sẽ được đưa lên **firebase.google.com** sau mỗi 15s dưới mức 20%.



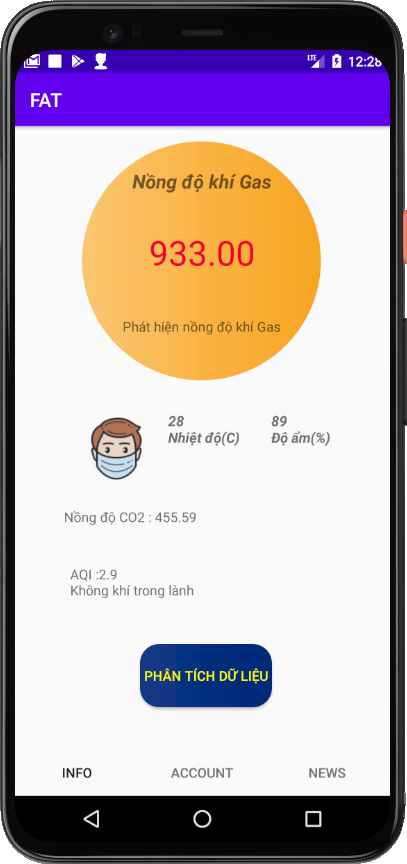
**Hình 4.7: Thiết bị báo khu vực an toàn không phát hiện rò rỉ khí gas.**

****

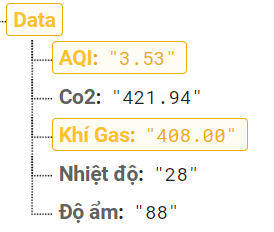
**Hình 4.8: Dữ liệu trên Realtime Database khí gas không bị rò.**

### **4.2.2 Hệ thống đang hoạt động có sự cố rò rỉ khí gas:**

Trong trường hợp khí gas bị rò rỉ ra môi trường, cảm biến sẽ phát hiện so sánh với mức an toàn là 20%. Nếu giá trị cảm biến vượt ngưỡng 20% thiết bị sẽ phát ra cảnh báo tại chỗ như đèn, còi và gọi điện, nhắn tin đến thuê bao đã cài đặt sẵn thông báo tình hình rò rỉ kịp thời để khắc phục sự cố. Dữ liệu đưa lên thingspeak.com sẽ vẽ đường vượt ngưỡng 20%.



**Hình 4.9: Thiết bị đang hoạt động khi có sự cố rò rỉ khí gas.**

****

**Hình 4.10: Dữ liệu trên Realtime Database phát hiện khí gas.**

# CHƯƠNG V: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1 Kết luận:

Sau quá trình nghiên cứu và tìm hiểu, tiến hành mô phỏng thử nghiệm thực tế và hoàn thành sản phẩm kịp với thời gian đề ra cùng với những yêu cầu ban đầu là đáp ứng đủ. Trong quá trình thực hiện sản phẩm đã có những kết quả thu được đáng kể.

- Thiết bị phát hiện khí gas, khí độc trong môi trường.

- Thiết bị phát hiện đo được nhiệt độ, độ ẩm trong môi trường.

- Giá trị khí gas, phát hiện khí độc, nhiệt độ và độ ẩm được so sánh với ngưỡng nguy hiểm liên tục và ổn định.

- Thông báo giá trị khí gas, khí độc, nhiệt độ và độ ầm qua App điện thoại.

- Thiết bị cảnh báo đúng với mức giá trị nguy hiểm.

- Gửi dữ liệu lên Website để theo dõi từ xa cũng như lưu trữ để khi cần sử dụng.

- Tín hiệu thiết bị hoạt động tốt.

## 5.2 Các điểm hạn chế của hệ thống:

- Hệ thống chưa có trạm xử lý tại chỗ khi có sự cố.

- Bị ảnh hưởng bởi tốc độ Internet.

- Phụ thuộc vào chính sách của Google.

- Dữ liệu lưu trên hệ thống tối đa 30 ngày.

## 5.3 Hướng phát triển:

- Chuông báo cháy: Được lắp đặt tại trung tâm xử lý, có chức năng phát tín hiệu báo động bằng âm thanh trong trường hợp có sự cố xảy ra, nhằm thông báo cho những người xung quanh biết để có phương án xử lý, di tản kịp thời.

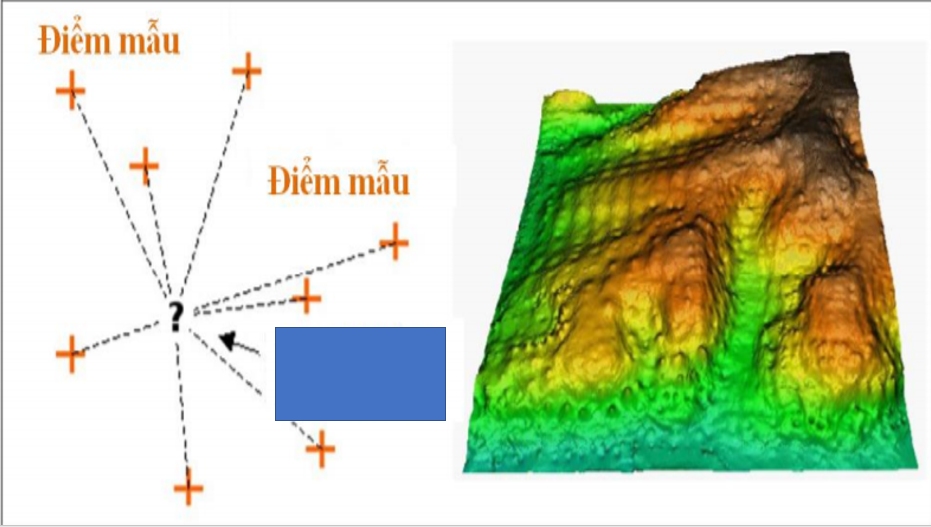
- Đèn: tượng trưng cho các trạm xử lý, có chức năng báo động và giúp người giám sát có thể xác định được vị trí xảy ra sự cố.

- Module Sim: được dùng để gửi tín hiệu thông báo bằng tin nhắn đến người quản lý nhằm đảm bảo tính khẩn cấp.

- Các biện pháp phòng chống được lắp sẵn gồm các đầu phun tự động, các quạt thông gió.

- Có thể thay đổi các vi điều khiển, linh kiện và thiết bị tốt hơn để tối ưu thời gian và độ chính xác của hệ thống hơn nữa (nếu có điều kiện).

* Ngoài ra, phát triển đề tài theo hướng machine learning nâng cấp từ việc xử lý và phát triển thành hướng phân tích dự đoán chất lượng không khí trong tương lai.

****

**Hình 5.1: Mẫu hướng phát triển cho hệ thống.**

# TÀI LIỆU THAM KHẢO

## 1. Sách tham khảo:

[1] Đoàn Thanh Đủ. *Đồ án Hệ thống giám sát và báo động khí gas*, 07/2018

[2] Trần Minh Tâm và Đỗ Thị Huê. *Thiết kế và thi công hệ thống cảnh báo, phòng chống hỏa hoạn và rò rỉ khí gas*, 6/2019

## 2. Trang Web tham khảo:

http://arduino.vn, “ Giới thiệu ESP8266 NodeMCU”

http://arduino.vn, “ Giới thiệu vắn tắt kít phát triển ESP8266 (chip nạp CP2102)”

http://arduino.vn/search/node/ESP8266%20k%E1%BA%BFt%20n%E1%BB%91i%20Internet

http://arduino.vn/bai-viet/899-huong-dan-su-dung-cam-bien-khi-gas-mq2-voi-arduino

<http://arduino.vn/bai-viet/1545-gioi-thieu-cam-bien-chat-luong-khong-khi-mq135>

http://arduino.vn/tutorial/1483-huong-dan-xay-dung-he-thong-giam-sat-qua-internet-don-gian

<https://esp8266.vn/nonos-sdk/basic/uart/> “Internet Of Things(IoT) với ESP8266”

<https://viblo.asia/p/lam-viec-voi-firebase-realtime-database-ZjlvalVXkqJ>

http://arduino.vn, “ Software Serial - Giao tiếp giữa Arduino và nhiều mạch Serial khác - Truyền tải trung gian giữa một mạch khác qua giao tiếp Serial”

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

# PHỤ LỤC

## 1. Phân chia công việc trong nhóm:

|  |  |  |
| --- | --- | --- |
| Tên Sinh viên thực hiện | Nội dung công việc phân công | Thời gian thực hiện |
| Lê Thanh Hiệp | Tìm hiểu, viết chương trình cho mô hình + App điều khiển trên thiết bị di động | 10/06/2021-10/07/2021 |
| Phạm Thị Ngọc Hân  Hồ Ngọc Giả | Tìm hiểu mạch, lắp mạch cho mô hình | 20/05/2021-20/06/2021 |
| Hồ Ngọc Giả | Viết báo cáo kết thúc học phần | 01/07/2021-27/07/2021 |
| Phạm Thị Ngọc Hân  Hồ Ngọc Giả | Làm Powerpoint thuyết trình | 20/07/2021-28/07/2021 |
| Lê Thanh Hiệp  Phạm Thị Ngọc Hân  Hồ Ngọc Giả | Thuyết trình báo cáo | 05/08/2021 |

**Bảng 7.1. Bảng phân chia công việc trong nhóm.**

## 2. Chương trình Phần cứng (Arduino IDE):

#include <MQ135.h>//Thêm thư viện

#include <DHT.h>//Thêm thư viện

#include <SoftwareSerial.h>//Thêm thư viện

#include <ESP8266WiFi.h>//Thêm thư viện

#include "FirebaseESP8266.h"//Thêm thư viện

#include <ArduinoJson.h>//Thêm thư viện

#define WIFI\_SSID "Bún Đậu Tỵ"// tên wifi

#define WIFI\_PASSWORD "BUNDAUTY"// Mật khẩu wifi

#define FIERBASE\_HOST "laptrinhnhung-9289f-default-rtdb.firebaseio.com"

#definec FIREBASE\_AUTH "F8UzDMZRiC1RkeRmydxbdhWQzVu8x36RC2AA

ihI2"

#define DHTTYPE DHT11

#define DHTPIN 2

DHT dht(DHTPIN, DHTTYPE);

FirebaseData firebaseData;

String path="/";

FirebaseJson json;

byte x=0;

String da="";

String gas;

String co2;

String clkk;

int dem = 0;

bool stringComplete = false;

int a = 1, b = 0, c = 0, d = 0;

long last = 0;

long last1 = 0;

long last2 = 0;

int vtt;

//cảm biến//

void setup(){

pinMode(D1, OUTPUT);//thiết đặt chân D1 là OUTPUT

pinMode(D2, OUTPUT);//thiết đặt chân D2 là OUTPUT

pinMode(D3, OUTPUT);//thiết đặt chân D3 là OUTPUT

pinMode(D4, OUTPUT);//thiết đặt chân D4 là OUTPUT

pinMode(D5, OUTPUT);//thiết đặt chân D5 là OUTPUT

pinMode(D6, OUTPUT);//thiết đặt chân D6 là OUTPUT

last = millis();

Serial.begin(9600);//Mở serial

dht.begin(); // Khởi động cảm biến

WiFi.begin(WIFI\_SSID,WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED) {

//Serial.print(".!.");

delay(500);// chờ 0.5s khi kết nối thất bại

}

//đọc dth

//Khởi tạo firebase

Firebase.begin(FIERBASE\_HOST, FIREBASE\_AUTH);

//Firebase.ReconnectWiFI(true);

if(!Firebase.beginStream(firebaseData, path))

{

//Serial.println("REASON: " + firebaseData.errorReason());

//Serial.println(".");

}

//Serial.print("connected: ");

//Serial.println(WiFi.localIP());

}

void loop()

{

guidata();

}

void guidata()

{

int doam = dht.readHumidity();// Đọc giá trị nhiệt độ từ cảm biến

int nhietdo = dht.readTemperature();// Đọc giá trị độ ẩm từ cảm biến

float g = analogRead(A0);//// Đọc giá trị khí gas

if(!isnan(doam))//kiểm tra dữ liệu doam có phải dạng số không

{

Firebase.setString(firebaseData, path+ "/Data/Độ ẩm", String(doam));

}

if(!isnan(nhietdo))//kiểm tra dữ liệu nhietdo có phải dạng số không

{

Firebase.setString(firebaseData, path+ "/Data/Nhiệt độ", String(nhietdo));

}

if(!isnan(g))//kiểm tra dữ liệu g có phải dạng số không

{

Firebase.setString(firebaseData, path+ "/Data/Khí Gas", String(g));

}

delay(500);

}

## 3. Chương trình Phần mềm (App):

* **Main:**

package vn.edu.tdmu.lethanhhiep.bcck;

import android.app.TabActivity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.TabHost;

import android.widget.Toast;

import vn.edu.tdmu.lethanhhiep.bcck.Service.ConnectionReceiver;

public class Main extends TabActivity {

private TabHost tabHost;

private TabHost.TabSpec tabSpec;

private Intent intent;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main2);

dangkynut();

ax();

}

private void dangkynut()

{

tabHost = (TabHost)findViewById(android.R.id.tabhost);

}

private void check()

{

boolean ret = ConnectionReceiver.isConnected();//Kiểm tra trạng thái kết nối Internet.

String ms;

if(ret)

{

ms = "The device has an Internet connection and can be done online";//Thiết bị có kết nối Internet và có thể thực hiện trực tuyến

}

else

{

ms = "The device does not have an Internet connection and can be performed offline";//Thiết bị không có kết nối Internet và có thể thực hiện ngoại tuyến

}

Toast.makeText(Main.this,ms,Toast.LENGTH\_SHORT).show();

}

private void ax()

{

tabSpec = tabHost.newTabSpec ("Info");

tabSpec.setIndicator ("Info");

intent = new Intent (this, tranghienthi.class);

tabSpec.setContent (intent);

tabHost.addTab (tabSpec);

tabSpec = tabHost.newTabSpec ("Account");

tabSpec.setIndicator ("Account");

intent = new Intent (this, trangcanhan.class);

tabSpec.setContent (intent);

tabHost.addTab (tabSpec);

tabSpec = tabHost.newTabSpec ("News");

tabSpec.setIndicator ("News");

intent = new Intent (this, tintuc.class);

tabSpec.setContent (intent);

tabHost.addTab (tabSpec);

tabHost.setCurrentTab(1);

}

@Override

protected void onStart()

{

check();

super.onStart();

}

private class sukiencuatoi implements View.OnClickListener

{

@Override

public void onClick(View view)

{

}

}

}

* **MainActivity:**

package vn.edu.tdmu.lethanhhiep.bcck;

import android.app.Activity;

import android.app.ProgressDialog;

import android.content.Intent;

import android.database.Cursor;

import android.os.Bundle;

import android.util.Log;

import android.view.KeyEvent;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

//Thêm thư viện CSDL Firebase.google.com//

import com.google.firebase.auth.AuthResult;

import com.google.firebase.auth.FirebaseAuth;

import vn.edu.tdmu.lethanhhiep.bcck.CSDL.sqlite;

import vn.edu.tdmu.lethanhhiep.bcck.Service.ConnectionReceiver;

public class MainActivity extends Activity {

private EditText edtName,edtmk;

private Button btndn,btndk;

public FirebaseAuth firebaseAuth;

public Intent intent;

static String tend,ten1;

public Cursor cursor;

public sqlite dl;

public int numBer = 0;

private ProgressDialog progressDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

dangkynut();

dangkysukien();

}

private void check()

{

boolean ret = ConnectionReceiver.isConnected();//Kiểm tra trạng thái kết nối Internet

String ms;

if (ret)

{

ms = "The device has an Internet connection and can be done online";//Thiết bị có kết nối Internet và có thể thực hiện trực tuyến

ten1 = "ok";//

}

else

{

ms = "The device has no Internet connection and can be performed offline";//Thiết bị không có kết nối Internet và có thể thực hiện ngoại tuyến

ten1 = "ko";

btndk.setEnabled (false);//Nút đăng kí ở trạng thái không thể nhấn.

}

Toast.makeText(MainActivity.this,ms,Toast.LENGTH\_SHORT).show();//Hiển thị thông báo

}

private void IsCheck()

{

try {

dl = new sqlite(MainActivity.this, "dulieunguoidung.sqlite", null, 1);

cursor = dl.truyvancoketqua("SELECT \* FROM nguoidung WHERE ten='" + edtName.getText().toString().trim() + "' AND matkhau='" + edtmk.getText().toString().trim() + "'");//Kiểm tra tài khoản và mật khẩu nhập vào với CSDL

if (cursor != null)

{

if(cursor.getCount() == 0)

{

Toast.makeText(MainActivity.this, "Username or password incorrect", Toast.LENGTH\_SHORT).show();//Thông báo: Tài khoản hoặc mật khẩu nhập vào chưa đúng

}

intent = new Intent(MainActivity.this, Main.class);

while (cursor.moveToNext())

{

tend = cursor.getString(0);

Toast.makeText(MainActivity.this, "Logged in successfully", Toast.LENGTH\_SHORT).show();//Thông báo: Đăng nhập thành công

edtName.setText("tomhumchinvn@gmail.com");

edtmk.setText("123456789");

startActivity(intent);

}

}

}

catch (Exception e)

{

Toast.makeText(MainActivity.this,"Login unsuccessful",Toast.LENGTH\_SHORT).show();//Thông báo: Đăng nhập không thành công

}

}

//Để xử lý các phím phần cứng và phím Quay lại//

@Override

public boolean dispatchKeyEvent(KeyEvent event)

{

Log.i("key pressed", String.valueOf(event.getKeyCode()));

return super.dispatchKeyEvent(event);

}

@Override

protected void onStart()

{

check();

super.onStart();

}

private void dangkynut()

{

edtName = findViewById(R.id.use);

edtmk = findViewById(R.id.pass);

btndn = findViewById(R.id.button);

btndk = findViewById(R.id.btndangkytk);

edtName.setOnKeyListener(new View.OnKeyListener()

{

@Override

public boolean onKey(View v, int keyCode, KeyEvent event)

{

if(event.getAction() == KeyEvent.ACTION\_UP)

{

if(edtName.getText().toString().trim().length() < 1)

{

btndn.setEnabled(false);

}

else

{

}

}

return false;

}

});

edtmk.setOnKeyListener(new View.OnKeyListener() {

@Override

public boolean onKey(View v, int keyCode, KeyEvent event)

{

if(event.getAction() == KeyEvent.ACTION\_UP)

{

if(edtmk.getText().toString().trim().length() < 1)

{

btndn.setEnabled(false);

}

else

{

btndn.setEnabled(true);

}

}

return false;

}

});

}

private void dangkysukien()

{

btndn.setOnClickListener(new sukiencuatoi());

btndk.setOnClickListener(new sukiencuatoi());

}

private void ax(String eMail)

{

firebaseAuth = FirebaseAuth.getInstance();// Xác thực Firebase để đăng nhập Email cho ứng dụng

//Bạn có thể gửi email đặt lại mật khẩu cho người dùng//

firebaseAuth.sendPasswordResetEmail(eMail).addOnCompleteListener(new OnCompleteListener<Void>() {

@Override

public void onComplete(@NonNull Task<Void> task)

{

if(task.isSuccessful())

{

Toast.makeText(MainActivity.this, "Please check your email for new password update", Toast.LENGTH\_SHORT).show();//Thông báo: Bạn vui lòng kiểm tra email cho việc đặt lại mật khẩu

}

}

});

}

private class sukiencuatoi implements View.OnClickListener

{

@Override

public void onClick(View view)

{

if(view.equals(btndn))

{

if(ten1.equals("ok"))

{

dangnhap();

}

if(ten1.equals("ko"))

{

IsCheck();

}

}

if(view.equals(btndk))

{

Intent intent = new Intent(MainActivity.this,Signup.class);

startActivity(intent);

}

}

}

private void dangnhap()

{

try {

progressDialog = new ProgressDialog(MainActivity.this);

progressDialog.setMessage("Please wait a moment");

progressDialog.show();

String ten = "tomhumchinvn@gmail.com";

String mk = "123456789";

firebaseAuth = FirebaseAuth.getInstance();// Xác thực Firebase để đăng nhập với Email và mật khẩu nhập vào cho ứng dụng

//Xác nhận kích hoạt đăng nhập với Email và mật khẩu nhập vào//

firebaseAuth.signInWithEmailAndPassword(ten.trim(), mk.trim()).addOnCompleteListener(MainActivity.this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task)

{

if (task.isSuccessful())

{

progressDialog.dismiss();

intent = new Intent(MainActivity.this,Main.class);

tend = firebaseAuth.getCurrentUser().getUid();//để lấy id người dùng của người dùng. Phương thức này trả về số thay vì id người dùng thực tế trong một số trường hợp

startActivity(intent);

} else

{

progressDialog.dismiss();

numBer++;

if(numBer == 3)

{

ax(edtName.getText().toString());

numBer = 0;

}

Toast.makeText(MainActivity.this, "Login failed", Toast.LENGTH\_SHORT).show();//Thông báo: Đăng nhập thất bại.

}

progressDialog.dismiss();

}

});

}

catch (Exception e)

{

progressDialog.dismiss();

Toast.makeText(MainActivity.this,"Login unsuccessful",Toast.LENGTH\_SHORT).show();//Thông báo: Kết nối không thành công.

}

}

}

* **tranghienthi:**

package vn.edu.tdmu.lethanhhiep.bcck;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import android.graphics.Color;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.TextView;

import android.widget.Toast;

//Thêm thư viện CSDL Firebase.google.com//

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;

public class tranghienthi extends AppCompatActivity {

TextView txttb,txtts,txtkg,txtcb,txtttco2,txtAQI,txtdam;

ImageView imgus;

Button btnmr;

FirebaseDatabase firebaseDatabase;

DatabaseReference databaseReference;

@Override

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_tranghienthi);

addcontrol();

addevent();

ax();

}

public void addcontrol()

{

txtcb = findViewById(R.id.txtcanhbao);

txtttco2 = findViewById(R.id.txtttco2);

txtkg = findViewById(R.id.txtkg);

txttb = findViewById(R.id.txttb);

txtts = findViewById(R.id.txtts);

imgus = findViewById(R.id.imguser);

txtAQI = findViewById(R.id.txtAQI);

txtdam = findViewById(R.id.txtdam);

btnmr = findViewById(R.id.btnmr);

}

public void ax()

{

firebaseDatabase = FirebaseDatabase.getInstance//Nhận phiên bản CSDL Firebase mặc định

databaseReference = firebaseDatabase.getReference("Data");//Nhận CSDL cho nút gốc CSDL là Data.

databaseReference.addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot snapshot) //Lấy dữ liệu thành công

{

txtkg.setTextColor(Color.parseColor(canhbao(snapshot.child("Khí Gas").getValue().toString())));

txtcb.setText(tb2(snapshot.child("Khí Gas").getValue().toString()));

txtkg.setText(snapshot.child("Khí Gas").getValue().toString());

txtAQI.setText(tb(snapshot.child("AQI").getValue().toString()));

txtttco2.setText("Nồng độ CO2: "+snapshot.child("Co2").getValue().toString());

String nhietdo = snapshot.child("Nhiệt độ").getValue().toString() +"\nNhiệt độ(C)";

String doam = snapshot.child("Độ ẩm").getValue().toString() + "\nĐộ ẩm(%)";

txtts.setText(nhietdo);

txtdam.setText(doam);

imgus.setBackgroundResource(td(snapshot.child("AQI").getValue().toString(),snapshot.child("Khí Gas").getValue().toString()));

}

@Override

public void onCancelled(@NonNull DatabaseError error) //Lấy dữ liệu xảy ra lỗi

{

}

});

}

public String canhbao(String tt)

{

float kq = Float.parseFloat(tt);//Kết quả đo được của cảm biến

String mau = kq >=700 ? "#f30031":kq >= 600 ? "#ec5f5f":"#ffffff";//Hiển thị Cảnh báo tùy theo mức độ giá trị đo được

return mau;

}

public String tb(String tt)

{

float kq = Float.parseFloat(tt);//Kết quả đo được của cảm biến

String k = kq > 300 ? "Mọi người nên ở trong nhà":kq >= 201 ? "Mọi người nên hạn chế ra ngoài":kq >= 51 ? "Những người nhạy cảm nên hạn chế ở bên ngoài":"Không khí trong lành";//Hiển thị Thông báo tùy theo mức độ giá trị đo được

String kq2 = "AQI :"+String.valueOf(kq) +"\n" + k;//Hiển thị chỉ số Chất lượng không khí

return kq2;

}

public int td(String aqi,String gas)

{

float k1 = Float.parseFloat(aqi);

float k2 = Float.parseFloat(gas);

int kq2 = k1 >= 600 ? 1:0;

int kq = k2 > 201 ? 1:0;

int kq3 = kq + kq2 >= 1 ? R.drawable.rattoite:R.drawable.userfuny;

return kq3;

}

public String tb2(String tt)

{

float kq = Float.parseFloat(tt);//Kết quả đo được của cảm biến

String k = kq > 700 ? "Phát hiện nồng độ khí Gas":kq >= 600 ? "Phát hiện nồng độ Alcohol":"An toàn";//Hiển thị thông báo tùy theo giá trị đo được

return k;

}

public String cbco2(String tt)

{

float kq = Float.parseFloat(tt);

String mau = kq >=700 ? "#f30031":kq >= 600 ? "#ec5f5f":"#f8ac32";

return mau;

}

public void addevent()

{

btnmr.setOnClickListener(new myevent());

}

public class myevent implements View.OnClickListener

{

@Override

public void onClick(View view)

{

if(view.equals(btnmr))

{

Toast.makeText(tranghienthi.this,"Hiện tại tính năng đang trong quá trình phát triển",Toast.LENGTH\_SHORT).show();

}

}

}

}

* **trangcanhan:**

package vn.edu.tdmu.lethanhhiep.bcck;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AlertDialog;

import androidx.appcompat.app.AppCompatActivity;

import android.app.Dialog;

import android.content.DialogInterface;

import android.content.Intent;

import android.database.Cursor;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

//Thêm thư viện CSDL Firebase.google.com//

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.FirebaseUser;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;

import java.util.ArrayList;

import java.util.HashMap;

import vn.edu.tdmu.lethanhhiep.bcck.CSDL.sqlite;

import vn.edu.tdmu.lethanhhiep.bcck.Model.User;

public class trangcanhan extends AppCompatActivity

{

private EditText edtten,edtmk,edtdc,edtns;

private Button btndy,btnluu;

public Intent intent;

public Cursor cursor;

private sqlite dl;

private int kk = 0;

public FirebaseDatabase firebaseDatabase;

public DatabaseReference databaseReference;

public FirebaseAuth firebaseAuth = FirebaseAuth.getInstance();////Nếu người dùng đã đăng nhập thành công, bạn có thể lấy dữ liệu tài khoản bằng phương pháp này.

public FirebaseUser firebaseUser = firebaseAuth.getCurrentUser();//Nếu người dùng đã đăng nhập thành công, bạn có thể lấy dữ liệu tài khoản bằng phương pháp này.

private ArrayList<User> arrayList;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_trangcanhan);

firebaseDatabase = FirebaseDatabase.getInstance();();//Nhận phiên bản CSDL Firebase mặc định

databaseReference = firebaseDatabase.getReference("dangnhap");//Nhận CSDL cho nút gốc CSDL là dangnhap.

dangkynut();

dangkysukien();

taodt();

ax();

}

private void dangkynut()

{

edtten = findViewById(R.id.edttencanhan);

edtmk = findViewById(R.id.edtmkcanhan);

edtdc = findViewById(R.id.edtdiachicanhan);

edtns = findViewById(R.id.edtngaysinhcanhan);

btndy = findViewById(R.id.btndycn);

btnluu = findViewById(R.id.btnluucanhan);

}

private void dangkysukien()

{

btnluu.setOnClickListener(new sukiencuatoi());

btndy.setOnClickListener(new sukiencuatoi());

}

private void sukiendong()

{

AlertDialog.Builder builder = new AlertDialog.Builder(trangcanhan.this);

builder.setTitle ("Notice");

builder.setMessage ("Do you want to exit");

builder.setPositiveButton ("Yes", new DialogInterface.OnClickListener () {

@Override

public void onClick (DialogInterface dialog, int which)

{

finish();

}

});

builder.setNegativeButton("No", new DialogInterface.OnClickListener () {

@Override

public void onClick (DialogInterface dialog, int which)

{

dialog.dismiss ();

}

});

Dialog dialog1 = builder.create();

dialog1.show();

}

@Override

public void onBackPressed()

{

super.onBackPressed();

}

private void taodt()

{

try {

String ssa = MainActivity.tend;

arrayList = new ArrayList<>();

dl = new sqlite(trangcanhan.this, "dulieunguoidung.sqlite", null, 1);

dl.truyvankhongtrakq("CREATE TABLE IF NOT EXISTS nguoidung(ID VARCHAR(50) PRIMARY KEY,ten VARCHAR(50),matkhau VARCHAR(100),ngaysinh VARCHAR(20),diachi VARCHAR(200))");

cursor = dl.truyvancoketqua("SELECT \* FROM nguoidung WHERE ID='" + ssa + "'");

if (cursor != null) {

while (cursor.moveToNext()) {

String ten = cursor.getString(1);

String mk = cursor.getString(2);

String ngaysinh = cursor.getString(3);

String diachi = cursor.getString(4);

arrayList.add(new User(ten, mk, ngaysinh, diachi));

kk = 4;

}

}

DoDuLieu();

}

catch (Exception e)

{

Toast.makeText(trangcanhan.this,"Error! An error occurred. Please try again later",Toast.LENGTH\_SHORT).show();

Intent intent = new Intent(trangcanhan.this,MainActivity.class);

startActivity(intent);

}

}

private void DoDuLieu()

{

try {

for (int i = 0; i < arrayList.size(); ++i) {

edtten.setText(arrayList.get(i).getHoten());

edtmk.setText(arrayList.get(i).getMatkhau());

edtns.setText(arrayList.get(i).getNgaysinh());

edtdc.setText(arrayList.get(i).getDiachi());

}

}

catch (Exception e)

{

Toast.makeText(trangcanhan.this,"Error! An error occurred. Please try again later",Toast.LENGTH\_SHORT).show();

intent = new Intent(trangcanhan.this,MainActivity.class);

startActivity(intent);

}

}

private void ax()

{

try {

arrayList = new ArrayList<>();

String id = MainActivity.tend;

databaseReference.child(id).addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot snapshot) {

String Ma = snapshot.getKey();

String Ten = snapshot.child("Tên").getValue().toString();

String Mk = snapshot.child("Mật khẩu").getValue().toString();

String diachi = snapshot.child("Địa chỉ").getValue().toString();

String ngaysinh = snapshot.child("Ngày sinh").getValue().toString();

if (kk == 0)

{

dl.truyvankhongtrakq("INSERT INTO nguoidung VALUES('" + Ma + "','" + Ten + "','" + Mk + "','" + ngaysinh + "','" + diachi + "')");

}

arrayList.add(new User(Ten, Mk, ngaysinh, diachi));

for (int i = 0; i < arrayList.size(); ++i) {

edtten.setText(arrayList.get(i).getHoten());

edtmk.setText(arrayList.get(i).getMatkhau());

edtns.setText(arrayList.get(i).getNgaysinh());

edtdc.setText(arrayList.get(i).getDiachi());

}

}

@Override

public void onCancelled(@NonNull DatabaseError error)

{

Toast.makeText(trangcanhan.this,"Please check the internet speed",Toast.LENGTH\_SHORT).show();

}

});

}

catch (Exception e)

{

Toast.makeText(trangcanhan.this,"Data is conflicting, please login again",Toast.LENGTH\_SHORT).show();

intent = new Intent(trangcanhan.this,MainActivity.class);

startActivity(intent);

}

}

private void capnhatdl()

{

dl = new sqlite(trangcanhan.this, "dulieunguoidung.sqlite", null, 1);

firebaseUser = firebaseAuth.getCurrentUser();

String ten = edtten.getText().toString();

String mk = edtmk.getText().toString();

String ns = edtns.getText().toString();

String dc = edtdc.getText().toString();

HashMap<String,Object> result = new HashMap<>();

result.put("Ngày sinh",ns);

result.put("Mật khẩu",mk);

result.put("Tên",ten);

result.put("Địa chỉ",dc);

databaseReference.child(MainActivity.tend).updateChildren(result);

firebaseUser.updatePassword(mk).addOnCompleteListener(new OnCompleteListener<Void>() {

@Override

public void onComplete(@NonNull Task<Void> task) {

if(task.isSuccessful())

{

Toast.makeText(trangcanhan.this,"Change password successfully",Toast.LENGTH\_SHORT).show();

dl.truyvankhongtrakq("UPDATE nguoidung SET ten='"+ten+"',matkhau = '"+mk+"',ngaysinh = '"+ns+"',diachi = '"+dc+"' WHERE ID = '"+MainActivity.tend+"'");

}

}

});

Toast.makeText(trangcanhan.this,ns,Toast.LENGTH\_SHORT).show();

}

private class sukiencuatoi implements View.OnClickListener

{

@Override

public void onClick(View view)

{

if(view.equals(btndy))

{

sukiendong();

}

if(view.equals(btnluu))

{

capnhatdl();

}

}

}

}

## 4. Chương trình cho Firebase.google.com:

#include <MQ135.h>//Thêm thư viện

#include <DHT.h>//Thêm thư viện

#include <SoftwareSerial.h>//Thêm thư viện

#include <ESP8266WiFi.h>//Thêm thư viện

#include "FirebaseESP8266.h"//Thêm thư viện

#include <ArduinoJson.h>//Thêm thư viện

#define WIFI\_SSID "Bún Đậu Tỵ"// tên wifi

#define WIFI\_PASSWORD "BUNDAUTY" // Mật khẩu wifi

#define PIN\_MQ135 A0 //Khai báo pin nối với chân AO

MQ135 mq135\_sensor = MQ135(PIN\_MQ135); //Khai báo đối tượng thư viện

#define FIERBASE\_HOST "laptrinhnhung-9289f-default-rtdb.firebaseio.com"

#define FIREBASE\_AUTH "F8UzDMZRiC1RKeRmydxbdhWQzVu8x36RC2AAihI2"

#define DHTTYPE DHT11

#define DHTPIN 2

DHT dht(DHTPIN, DHTTYPE);

FirebaseData firebaseData;

String path="/";

FirebaseJson json;

byte x=0;

String da="";

String gas;

String co2;

String clkk;

int dem = 0;

bool stringComplete = false;

int a = 1, b = 0, c = 0, d = 0;

long last = 0;

long last1 = 0;

long last2 = 0;

int vtt;

//cảm biến

//

void setup(){

pinMode(D1, OUTPUT);//thiết đặt chân D1 là OUTPUT

pinMode(D2, OUTPUT);//thiết đặt chân D2 là OUTPUT

pinMode(D3, OUTPUT);//thiết đặt chân D3 là OUTPUT

pinMode(D4, OUTPUT);//thiết đặt chân D4 là OUTPUT

pinMode(D5, OUTPUT);//thiết đặt chân D5 là OUTPUT

pinMode(D6, OUTPUT);//thiết đặt chân D6 là OUTPUT

last = millis();

Serial.begin(9600);//Mở serial

dht.begin(); // Khởi động cảm biến

WiFi.begin(WIFI\_SSID,WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED)

{

//Serial.print(".!.");

delay(500);// chờ 0.5s khi kết nối thất bại

}

//đọc dth

//Khởi tạo firebase

Firebase.begin(FIERBASE\_HOST, FIREBASE\_AUTH);

//Firebase.ReconnectWiFI(true);

if(!Firebase.beginStream(firebaseData, path))

{

//Serial.println("REASON: " + firebaseData.errorReason());

//Serial.println(".");

}

//Serial.print("connected: ");

//Serial.println(WiFi.localIP());

}

void loop()

{

guidata();

}

void guidata()

{

float rzero = mq135\_sensor.getRZero();

float clkk = mq135\_sensor.getPPM();//Đọc giá trị ppm

if(!isnan(rzero))//kiểm tra dữ liệu rzero có phải dạng số không

{

Firebase.setString(firebaseData, path+ "/Data/Co2", String(rzero));

}

if(!isnan(clkk))//kiểm tra dữ liệu clkk có phải dạng số không

{

Firebase.setString(firebaseData, path+ "/Data/AQI", String(clkk));

}

delay(500);

}