

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**NIÊN LUẬN NGÀNH
TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

Đề tài

**HỆ THỐNG THU THẬP DỮ LIỆU CHECK IN DỰA
TRÊN BLUETOOTH**

**Sinh viên thực hiện :
Nguyễn Thanh Huy
Khóa : K43**

Cần Thơ, 06/2021

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**NIÊN LUẬN
NGÀNH TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

Đề tài

**HỆ THỐNG THU THẬP DỮ LIỆU CHECK IN DỰA
TRÊN BLUETOOTH**

**Giáo viên hướng dẫn:
TS. Ngô Bá Hùng**

Khóa : K43

**Sinh viên thực hiện:
Nguyễn Thanh Huy**

Cần Thơ, 06/2021

[illegible]

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Ngô Bá Hùng – Giảng viên hướng dẫn bộ môn Niên Luận Cơ Sở đã giành thời gian quý báu để hướng dẫn cho em trong môn học này. Cảm ơn thầy, đã tạo điều kiện và cung cấp những kiến thức quan trọng để học tập và thực hiện đề tài một cách tốt nhất.

Thông qua quá trình thực hiện niên luận, em phần nào củng cố và tích lũy được những kiến thức về lập trình ngôn ngữ Python trên Raspberry Pi. Mặc dù, đã cố gắng hết sức trong tất cả quá trình từ học tập cho đến thực hiện đề tài này. Nhưng em đã không thể tránh khỏi những sai sót nhất định. Em rất mong được sự thông cảm, bỏ qua và góp ý tận tình từ Thầy và các bạn.

Em xin chân thành cảm ơn !

Cần Thơ, ngày 06 tháng 06 năm 2021

Người viết

Nguyễn Thanh Huy

MỤC LỤC

PHẦN I. GIỚI THIỆU	1
1. Đặt vấn đề.....	1
2. Mục tiêu đề tài.....	1
2.1. Đối tượng nghiên cứu.....	2
2.2. Nội dung nghiên cứu	2
PHẦN II. NỘI DUNG	3
Chương 1. Mô Tả Bài Toán	3
1.1. Đặt tả yêu cầu	3
1.2. Đối tượng nghiên cứu.....	3
1.3. Công nghệ Bluetooth.....	4
1.3.1. Tổng quan.....	4
1.3.2. Đặc điểm.....	4
1.3.3. Khái niệm dùng trong công nghệ Bluetooth	5
1.3.3.1. Master Unit.....	5
1.3.3.2. Slaver Unit.....	5
1.3.3.3. Piconet	6
1.3.4. Các chế độ kết nối	7
1.3.5. Trạng thái của thiết bị Bluetooth.....	7
1.3.6. Các tầng giao thức	8
1.3.6.1. Bluetooth Radio.....	8
1.3.6.2. Baseband	8
1.3.6.3. Link Manager Protocol.....	9
1.3.6.4. Host Controller Interface.....	9
1.3.6.5. Logical Link Control and Adaptation Protocol	9
1.3.6.6. Radio Frequency Communication (RFCOMM).....	9
1.3.6.7. Service Discovery Protocol – SDP.....	10
1.3.6.8. Cấu trúc gói tin	10
1.4. Module Bluetooth tích hợp trên Smartphone	11
1.4.1. Đặc điểm.....	11
1.5. Raspberry Pi 4B.....	12
1.5.1. Tổng quan.....	12
1.5.2. Phần cứng	12
1.5.3. Hệ điều hành tương thích	13
1.6. API Bluetooth trên lập trình Android Studio	14
1.6.1. Tổng quan.....	14

1.6.2. Đặc điểm.....	14
1.6.3. Class Interface	15
1.7. Thư viện Pybluez.....	17
1.7.1. Tổng quan.....	17
1.7.2. Chọn giao tiếp với thiết bị.....	17
1.7.3. Giao thức kết nối	18
1.7.3.1. Giao thức RFCOMM + TCP	19
1.7.3.2. Service Discovery Protocol	19
1.7.3.3. Thư viện Pybluez.....	20
Chương 2. Thiết Kế Giải Pháp	21
1. Kiến trúc hệ thống:	21
1.1. Mô hình tổng thể các thành phần trong hệ thống	21
1.2. Mô hình Use case Hệ thống thu thập dữ liệu check in dựa trên Bluetooth.....	22
1.3. Mô hình Sequence Hệ thống thu thập dữ liệu check in dựa trên Bluetooth.....	23
1.4. Data Flow Diagram Hệ thống thu thập dữ liệu check in dựa trên Bluetooth.....	24
1.5. Data Flow Diagram đăng ký active bluetooth lần đầu	25
2. Thiết kế giải thuật xử lý	26
2.1. Trên Raspberry Pi 4B	26
2.2. Trên Android Studio Lập trình giao diện mức thấp trên MainActivity.java.....	28
2.3. Lập trình giao diện mức thấp trên CommsActivity.java	30
Chương 3. Cài Đặt Giải Pháp	32
3.1. Cài đặt và quản trị hệ điều hành Raspbian trên Raspberry Pi	32
3.2. Thiết lập môi trường phát triển ứng dụng Pybluez	32
3.3. Cấp quyền cho ứng dụng Bluetooth_Checkin trên Android Studio.....	32
3.4. Khởi chạy ứng dụng Bluetooth_checkin trên Smartphone	33
3.5. Kiểm tra kết quả	36
3.5.1 Trên Hệ thống thu thập dữ liệu check in dựa trên Bluetooth	36
3.5.2. Trên ứng dụng Bluetooth_checkin của Smartphone	37
Chương 4. Kiểm Thử Và Đánh Giá	38
1. Kiểm thử.....	38
2. Đánh giá	38
PHẦN III. KẾT LUẬN	39
1. Kết quả đạt được	39
2. Hướng phát triển.....	39
TÀI LIỆU THAM KHẢO	40

DANH MỤC HÌNH

Hình 1. Piconet gồm 1 Slave	6
Hình 2. Piconet gồm nhiều Slave	6
Hình 3. Các tầng giao thức Bluetooth	8
Hình 4. Cấu trúc gói tin Bluetooth	10
Hình 5. Raspberry Pi 4B.....	12
Hình 6. Hệ điều hành Raspbian.....	13
Hình 8. Mô hình Bluetooth Check_in	21
Hình 9. Mô hình Use case	22
Hình 10. Mô hình Sequence	23
Hình 11. Data Flow Diagram	24
Hình 12. Data flow diagram đăng ký bluetooth lần đầu	25
Hình 13. Giao diện chính của ứng dụng Bluetooth_Checkin	33
Hình 14. Danh sách device bluetooth.....	34
Hình 15. Giao diện người dùng nhập dữ liệu check in – check out.....	35
Hình 16. Dữ liệu file log ghi nhận được	36
Hình 17. Thông báo check in thành công.....	37

PHẦN I. GIỚI THIỆU

1. Đặt vấn đề

Với sự bùng nổ của cuộc cách mạng công nghiệp 4.0, đi cùng với sự phát triển nhanh chóng tốc độ của Internet hiện nay, chúng ta đã thực hiện được nhiều công việc với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Chính vì điều này, đã thúc đẩy sự khai sinh ra mạng lưới thiết bị kết nối không dây ra đời làm thay đổi đáng kể bộ mặt thế giới. Kỹ thuật không dây phục vụ rất nhiều nhu cầu khác nhau của con người, từ làm việc, học tập đến giải trí như chơi game, xem phim, nghe nhạc... Kỹ thuật không dây đã đưa ra nhiều chuẩn với đặc điểm kỹ thuật khác nhau như WLAN với chuẩn 802.11, Lora, Zigbee, Bluetooth... Mỗi chuẩn kỹ thuật đều có những ưu, nhược điểm khác nhau, và Bluetooth đang đang dần nổi lên là kỹ thuật không dây tầm ngắn và hầu hết các chiếc Smartphone và các máy tính nhúng hiện nay đều trang bị module Bluetooth. Cùng với sự gia tăng lây nhiễm đại dịch toàn cầu(Covid-19) có thể lây bệnh khi tiếp xúc trực tiếp hoặc gián tiếp với người nhiễm bệnh. Chính vì điều đó chúng ta cần truy vết người nghi nhiễm bệnh, nên chúng ta cần có thiết bị ghi nhận lịch sử ra vào cty, bệnh viện cũng như nơi công cộng để kiểm tra xem lịch sử đi lại của bệnh nhân, nhằm cách ly đúng người bệnh và ngăn ngừa sự lây lan bệnh ra cộng đồng. Với sự phổ biến cũng như tiện ích của Smartphone, nên hầu như mọi người đều có một hoặc hai chiếc smartphone có trang bị công nghệ Bluetooth. Xuất phát từ lý do trên em thực hiện đề tài “HỆ THỐNG THU THẬP DỮ LIỆU CHECK IN DỰA TRÊN BLUETOOTH”.

2. Mục tiêu đề tài

Chúng ta xây dựng hệ thống thu thập dữ liệu từ những thiết bị Smartphone có trang bị công nghệ Bluetooth từ đó kết nối tới một Server lắng nghe trên các cổng được mở. Khi người dùng bước vào vùng phủ sóng của Server thì người dùng sẽ nhập thông tin cần thiết vào ứng dụng và gửi đi, và Server có trách nhiệm thu thập dữ liệu người dùng và lưu lại lịch sử check in.

Cách vận hành và quản lý đơn giản, khả năng kết nối nhanh chóng, tính bảo mật cao dựa trên công nghệ Bluetooth, và không làm thất thoát dữ liệu trên đường truyền không dây, nhằm tránh sai sót kết quả và tính sẵn dùng cho người dùng và quản lý kết quả thuận tiện.

2.1. Đối tượng nghiên cứu

Với nhu cầu của các cơ quan, cty, doanh nghiệp về ghi nhận lịch sử ra vào nhằm để kiểm soát, khả năng tiếp xúc lây nhiễm bệnh Covid-19 để đưa đi cách ly nhanh nhất có thể, để có thể làm giúp cho chủ sở hữu giảm thiểu rủi ro lây lan bệnh. Vì vậy ta cần tập chung giải pháp lưu trữ dữ liệu check in thường xuyên, đọc dữ liệu từ các thiết bị Smartphone lên Raspberry Server và xử lý rồi lưu file log và cập nhật liên tục.

Tìm hiểu về công nghệ Bluetooth bao gồm:

Trạng thái, chế độ và các bước kết nối.

Các tầng giao thức.

Cấu trúc gói tin.

Ứng dụng và ưu nhược điểm.

Tìm hiểu về Raspbrry Pi 4B:

Cấu hình phần cứng, phần mềm.

Chế độ kết nối Bluetooth trên Raspbian.

Cài đặt thư viện pybluez hỗ trợ cho module Bluetooth.

Tìm hiểu lập trình ứng dụng trên Android Studio:

Thư viện hỗ trợ Bluetooth trên Android Studio.

Lập trình giao diện mức thấp và giao diện mức cao.

Tìm hiểu và lập trình Bluetooth bằng thư viện Pybluez.

Thư viện hỗ trợ trên Pybluez.

Tạo Bluetooth Server bằng giao thức RFCOMM.

2.2. Nội dung nghiên cứu

- Tìm hiểu và cài đặt biến môi trường trên Raspberry Pi.
- Tìm hiểu và lập trình Bluetooth trên Android Studio.
- Thu thập dữ liệu từ thiết bị smartphone sau đó xử lý và lưu trữ dữ liệu trên Raspberry Pi.
- Thu thập dữ liệu từ Raspberry Pi sau đó xử lý trên ứng dụng Bluetooth_Checkin.

PHẦN II. NỘI DUNG

Chương 1. Mô Tả Bài Toán

1.1. Đặt tả yêu cầu

Câu hỏi đặt ra, chúng ta làm thế nào để xây dựng hệ thống thu thập dữ liệu check in từ những thiết bị Smartphone, dùng thư viện gì để giao tiếp cũng như nhập xuất tín hiệu Bluetooth giữa thiết bị Smartphone với Raspberry Pi. Để làm được những điều đó chúng ta cần nghiên cứu về mặt lý thuyết trên thư viện Pybluez, lập trình mức thấp trên Android Studio, máy tính nhúng Raspberry Pi, các module Bluetooth được trang bị trên Raspberry Pi và Smartphone.

1.2. Đối tượng nghiên cứu

Với nhu cầu của con người về các hệ thống check in ra vào trong cty, cửa hàng, bệnh viện, nơi công cộng...Nhất là trong năm đại dịch Covid-19 nên rất cần một hệ thống check in số lượng người ra vào, thông tin lịch khai báo, và điều cần thiết là tự động hóa nhằm tránh lây bệnh và tiện lợi cũng như dữ liệu luôn được cập nhật nhanh chóng và chính xác. Vì vậy ta cần tập chung giải pháp thiết lập giao thức kết nối Bluetooth trên Smartphone và Raspberry Pi, đọc dữ liệu từ Smartphone và ngược lại, lập trình Mobile App trên Smartphone, lập trình trên Raspberry Pi và xử lý rồi lưu dữ liệu vào file log liên tục.

Tìm hiểu về công nghệ Bluetooth: Đặc điểm khái niệm dùng trong Bluetooth, các chế độ và trạng thái kết nối, các tầng giao thức.

Tìm hiểu về module Bluetooth trên Smartphone bao gồm: Chuẩn kết nối Bluetooth.

Tìm hiểu về máy tính nhúng Raspberry Pi bao gồm: Module Bluetooth được trang bị trên Raspberry Pi, hệ điều hành tương thích.

Tìm hiểu về API Bluetooth trên lập trình Android Studio bao gồm: Bluetooth kết nối xuất tính hiệu, phương thức kết nối, giao thức kết nối Bluetooth trên Smartphone.

Tìm hiểu về thư viện Pybluez bao gồm: Giao thức kết nối Bluetooth, các chức năng hỗ trợ của thư viện Pybluez.

1.3. Công nghệ Bluetooth

1.3.1. Tổng quan

Bluetooth là công nghệ không dây cho phép các thiết bị điện, điện tử giao tiếp với nhau trong khoảng cách ngắn bằng sóng vô tuyến qua băng tần chung ISM(Industrial, Scientific, Medical) trong dãy tần 2.40 – 2.48 GHz. Đây là dãy băng tần không cần đăng ký, được dành riêng để dùng cho các thiết bị không dây trong công nghiệp, khoa học, y tế... Bluetooth được thiết kế nhằm mục đích thay thế dây cáp, kết nối vô tuyến giữa các thiết bị điện tử lại với nhau một cách thuận lợi với giá thành rẻ. Khi kích hoạt, Bluetooth có thể tự động định vị những thiết bị khác có chung công nghệ trong vùng phủ sóng xung quanh và bắt đầu kết nối với nhau. Nó được định hướng sử dụng cho việc truyền dữ liệu, hình ảnh lẫn tiếng nói.

1.3.2. Đặc điểm

- Tiêu thụ năng lượng thấp, cho phép ứng dụng được cho nhiều loại thiết bị, bao gồm cả thiết bị cầm tay.
- Giá thành thấp.
- Khoảng cách giao tiếp cho phép: Giữa hai thiết bị đầu cuối có thể lên đến 10m ngoài trời và ở môi trường có vật cản, khoảng cách các thiết bị đầu cuối và Access point có thể lên tới 100m và môi trường có vật cản là 30m.
- Bluetooth sử dụng dãy băng tần không cần đăng ký 2.4 GHz trên băng tần ISM. Tốc độ truyền tải dữ liệu có thể đạt tối đa 1 Mbps (do sử dụng tần số cao) mà các thiết bị không cần phải thấy trực tiếp nhau (light-of-sightrequirements).
- Dễ dàng trong việc phát triển ứng dụng: Bluetooth kết nối một ứng dụng này với một ứng dụng hoặc phần mềm khác thông qua các chuẩn “Bluetooth profiles”, do đó có thể độc lập về phần cứng cũng như hệ điều hành sử dụng.
- Bluetooth được dùng trong giao tiếp dữ liệu và âm thanh: Có 3 kênh để truyền âm thanh và 7 kênh để truyền dữ liệu trong một mạng cá nhân.
- An toàn và bảo mật: Được tích hợp với sự xác nhận và mã hóa(build in authentication encryption).

1.3.3. Khái niệm dùng trong công nghệ Bluetooth

1.3.3.1. Master Unit

Là thiết bị duy nhất trong 1 mạng Piconet, Master thiết lập đồng hồ đếm xung và kiểu bước nhảy (hopping) để đồng bộ tất cả các thiết bị trong cùng Piconet mà nó đang quản lý, thường là thiết bị đầu tiên chuyển dữ liệu. Master cũng quyết định số kênh truyền thông. Mỗi Piconet có một kiểu hopping duy nhất.

1.3.3.2. Slaver Unit

Là tất cả các thiết bị còn lại trong Piconet, 1 thiết bị không là Master thì phải là Slave. Tối đa 7 Slave dạng Active và 255 Slave Parked(Inactive) trong 1 Piconet.

Có 3 dạng Slave trong một Piconet:

- + Active: Slave hoạt động, có khả năng trao đổi thông tin với Master và các Slave Active khác trong Piconet. Các thiết bị ở trạng thái này được phân biệt thông qua 1 địa chỉ MAC (Media Access Control) hay AMA (Active Member Address) – đó là con số gồm 3 bit. Nên trong 1 Piconet có tối đa 8 thiết bị ở trong trạng thái này (1 cho Master và 7 cho Slave).

- + Standby: Standby là một dạng inactive, thiết bị trong trạng thái này không trao đổi dữ liệu, sóng radio không có tác động lên, công suất giảm đến tối thiểu để tiết kiệm năng lượng, thiết bị không có khả năng dò được bất cứ mã truy cập nào. Có thể coi là những thiết bị nằm ngoài vùng kiểm soát của Master.

- + Parked: Là một dạng inactive, chỉ 1 thiết bị trong 1 Piconet thường xuyên được đồng bộ với Piconet, nhưng không có 1 địa chỉ MAC. Chúng như ở trạng thái “ngủ” và sẽ được Master gọi dậy bằng tín hiệu “beacon” (tín hiệu báo hiệu). Các thiết bị ở trạng thái Packed được đánh địa chỉ trong qua địa chỉ PMA (Packed Member Address). Đây là con số 8 bits để phân biệt các Packed Slave với nhau và tối đa 255 thiết bị ở trạng thái này trong 1 Piconet.

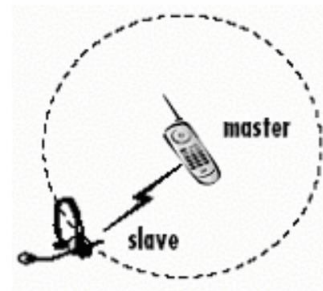
1.3.3.3. Piconet

Piconet là tập hợp các thiết bị được kết nối thông qua kỹ thuật Bluetooth theo mô hình Ad-Hoc (đây là kiểu mạng được thiết lập cho nhu cầu truyền dữ liệu hiện hành và tức thời, tốc độ nhanh và kết nối sẽ tự động hủy sau khi truyền xong). Trong 1 Piconet thì chỉ có 1 thiết bị là Master. Đây thường là thiết bị đầu tiên tạo kết nối, nó có vai trò quyết định số kênh truyền thông và thực hiện đồng bộ giữa các thành phần trong Piconet, các thiết bị còn lại là Slave, đó là các thiết bị gửi yêu cầu đến Master.

Lưu ý rằng, 2 Slave muốn thực hiện liên lạc phải thông qua Master bởi chúng không bao giờ kết nối trực tiếp với nhau, Master sẽ đồng bộ các Slave về thời gian và tần số. Trong 1 Piconet có tối đa 6 Slave và 1 Master đang hoạt động tại 1 thời điểm.

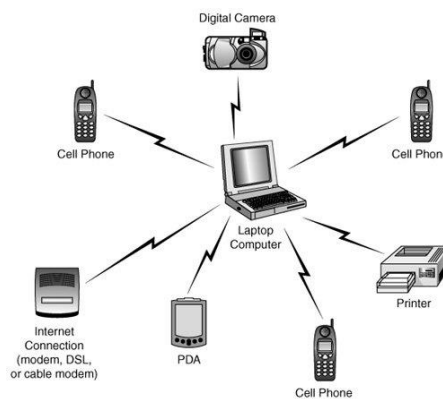
Các mô hình Piconet:

+ Piconet chỉ có 1 Slave:



Hình 1. Piconet gồm 1 Slave

+ Piconet gồm nhiều Slave:



Hình 2. Piconet gồm nhiều Slave

1.3.4. Các chế độ kết nối

Active mode: Trong chế độ này, thiết bị Bluetooth tham gia vào hoạt động của mạng. Thiết bị Master sẽ điều phối lưu lượng và đồng bộ hóa cho các thiết bị Slave.

Sniff mode: là 1 chế độ tiết kiệm năng lượng của thiết bị đang ở trạng thái active. Ở Sniff mode, thiết bị slave lắng nghe tín hiệu từ mạng với tần số giảm hay nói cách khác là giảm công suất. Tần số này phụ thuộc vào tham số của ứng dụng. Đây là chế độ ít tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.

Hold mode: Là 1 chế độ tiết kiệm năng lượng của thiết bị vẫn còn trong trạng thái active. Master có thể đặt chế độ Hold mode cho slave của mình. Các thiết bị có thể trao đổi dữ liệu ngay lập tức ngay khi thoát khỏi chế độ Hold mode. Đây là chế độ tiết kiệm năng lượng.

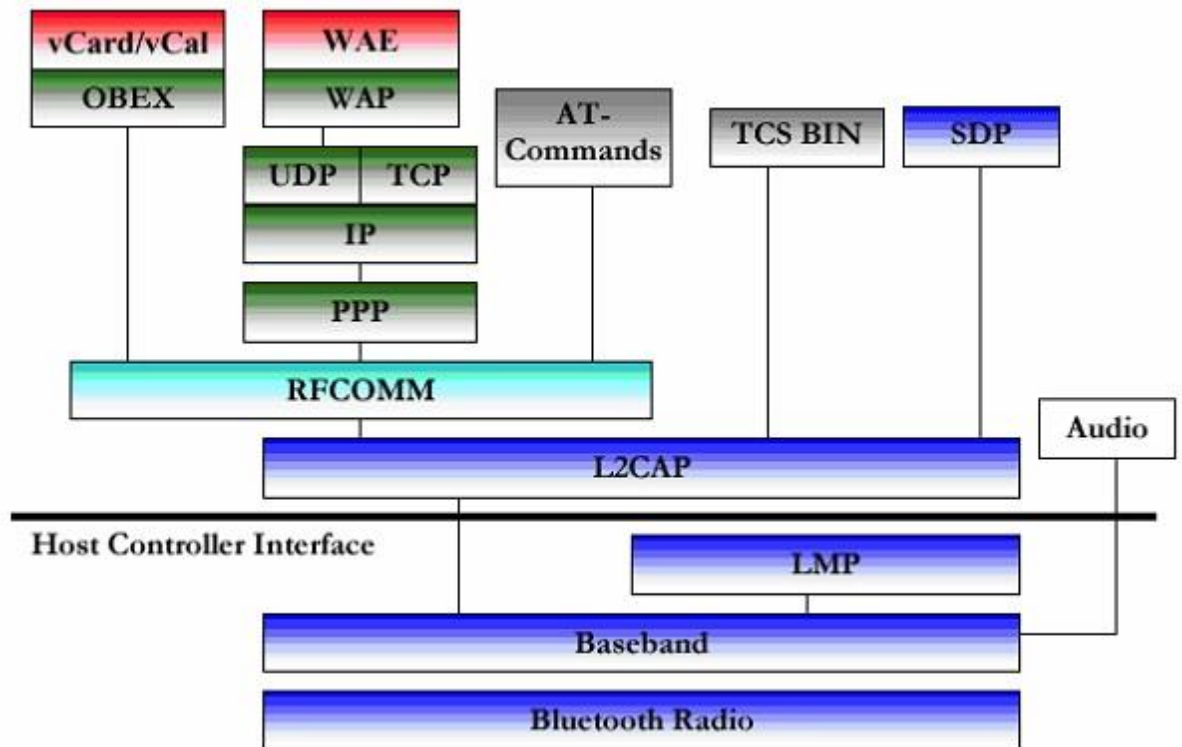
Park mode: Là chế độ tiết kiệm năng lượng của thiết bị vẫn còn trong mạng nhưng không tham gia vào quá trình trao đổi dữ liệu (inactive). Thiết bị ở chế độ Park mode bỏ địa chỉ MAC, chỉ lắng nghe tín hiệu đồng bộ hóa và thông điệp broadcast của Master. Đây là chế độ tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.

1.3.5. Trạng thái của thiết bị Bluetooth

Có 4 trạng thái chính của 1 thiết bị Bluetooth trong 1 Piconet:

- Inquiring device (inquiring mode): Thiết bị đang phát tín hiệu tìm thiết bị Bluetooth khác.
- Inquiry scanning device (inquiry scan mode): Thiết bị nhận tín hiệu inquiry của thiết bị đang thực hiện inquiry và trả lời.
- Paging device (page mode): Thiết bị phát tín hiệu yêu cầu kết nối với các inquiry khác.
- Page scanning device (page scan mode): Thiết bị nhận yêu cầu kết nối từ paging device và trả lời.

1.3.6. Các tầng giao thức



Hình 3. Các tầng giao thức Bluetooth

1.3.6.1. Bluetooth Radio

- Tầng Bluetooth Radio là tầng thấp nhất trong lớp giao thức.
- Định nghĩa những yêu cầu cho bộ phận thu phát sóng hoạt động ở tần số 2.4GHz ISM (là băng tần không cần đăng ký dành riêng cho công nghiệp, khoa học và y tế...).
- Sóng radio của Bluetooth được truyền bằng cách nhảy tần số, nghĩa là mọi packet được truyền trên những tần số khác nhau.
- Tốc độ nhảy nhanh giúp tránh nhiễu tốt.

1.3.6.2. Baseband

- Baseband nằm ở tầng vật lý của Bluetooth.
- Quản lý những kênh truyền và liên kết vật lý tách biệt khỏi các dịch vụ khác như sửa lỗi, chọn bước nhảy và bảo mật.
- Baseband Protocol được cài đặt như là một Link Controller, cùng với Link Manager thực hiện những công việc ở mức thấp như kết nối, quản lý năng lượng.
- Việc quản lý các kết nối đồng hồ và không đồng hồ, các gói tin, thực hiện tìm kiếm và yêu cầu kết nối các thiết bị Bluetooth khác.

1.3.6.3. Link Manager Protocol

- Link Manager thực hiện việc thiết lập kênh truyền, xác nhận hợp lệ và cấu hình kênh truyền.
- Tìm kiếm những Link Manager khác và giao tiếp với chúng thông qua Link Manager Protocol.
- Link Manager dùng những dịch vụ do tầng Link Controller cung cấp để thực hiện vai trò của mình.
- Các lệnh Link Manager Protocol bao gồm các PDU (Protocol Data Unit) được gửi từ thiết bị này sang thiết bị khác.

1.3.6.4. Host Controller Interface

- Cung cấp giao diện cho phép các tầng bên trên điều khiển Baseband và Link Manager, đồng thời cho phép truy cập đến trạng thái của phần cứng và các thanh ghi điều khiển.
- Host Controller Interface tồn tại trong 3 phần: Host – Transport layer – Host controller.
- Mỗi phần đóng góp một vai trò khác nhau trong hệ thống Host Controller Interface.

1.3.6.5. Logical Link Control and Adaptation Protocol

- Nằm trên giao thức băng tần cơ sở (Baseband Protocol) và nằm ở tầng Data link.
- L2CAP cung cấp dịch vụ hướng kết nối và phi kết nối cho các tầng giao thức bên trên.
- L2CAP có khả năng phân kênh (multiplexing), phân đoạn (segmentation) và tái tổ hợp (reassembly operation).
- L2CAP cho phép các giao thức ở các tầng trên và các ứng dụng truyền và nhận dữ liệu.
- Mỗi gói dữ liệu của L2CAP tối đa 64 Kbytes.

1.3.6.6. Radio Frequency Communication (RFCOMM)

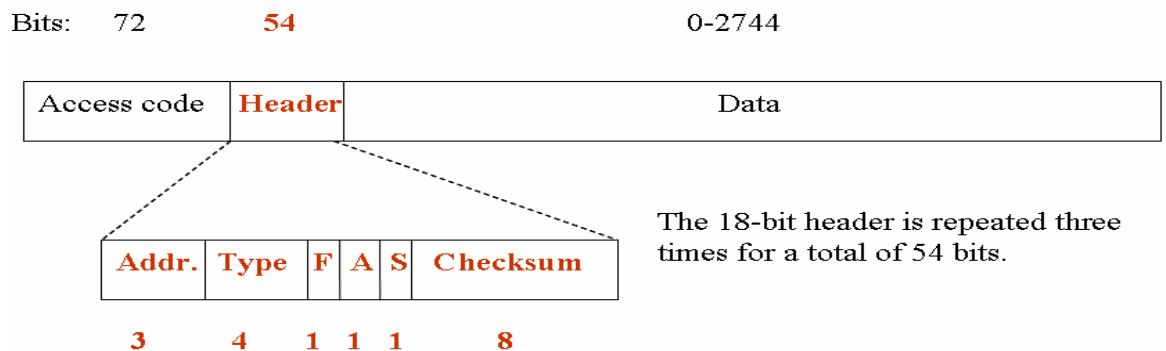
- Giao thức RFCOMM cho phép giả lập cổng serial thông qua giao thức L2CAP.
- RFCOMM dựa trên chuẩn ETSI TS 07.10. Chỉ có 1 phần của chuẩn TS 07.10 được dùng và chỉnh sửa cho phù hợp với Bluetooth.
- RFCOMM hỗ trợ tối đa 60 kết nối. Một kết nối bao gồm 2 ứng dụng chạy trên 2 thiết bị riêng biệt.
 - + Thiết bị 1: Thiết bị đầu cuối như máy tính, máy in,...
 - + Thiết bị 2: Thiết bị truyền dữ liệu như modem.

1.3.6.7. Service Discovery Protocol – SDP

- SDP cho phép các ứng dụng tìm kiếm những dịch vụ và thuộc tính của dịch vụ có trong thiết bị.
- SDP dùng mô hình request/response với mỗi thao tác bao gồm 1 request protocol data unit (PDU) và 1 response PDU.
- SDP có 3 dịch vụ chính:
 - + Service Record: Là nơi chứa các thuộc tính của dịch vụ.
 - + Service Attribute: Mô tả thuộc tính của dịch vụ.
 - + Service Class: Cung cấp các định nghĩa cho các thuộc tính trong Record.

1.3.6.8. Cấu trúc gói tin

- Mỗi packet chứa 3 phần: Access Code (Mã truy cập), Header, Payload.
- + Access code: Gồm 72 bits, dùng trong việc đồ bộ dữ liệu, định danh, báo hiệu.
- + Header có 54 bits mang tác dụng định danh.
- + Payload: Là phần chứa dữ liệu đi, có thể thay đổi từ 0 tới 2744 bit/packet. Payload có thể là dữ liệu voice hoặc data.



Hình 4. Cấu trúc gói tin Bluetooth

1.4. Module Bluetooth tích hợp trên Smartphone

1.4.1. Đặc điểm

Năm 2016, Bluetooth ra mắt truyền thông phiên bản 5.0 với hàng loạt sự cải tiến. Và sau đó Bluetooth 5.0 được tích hợp hầu hết lên những chiếc Smartphone và nhiều thiết bị điện tử khác.

Các bước cải tiến của Bluetooth 5.0 gồm: Tầm phủ sóng xa hơn lên đến 300m (trong điều kiện lý tưởng), gấp 4 lần so với Bluetooth 4.0. Còn trên thực tế Bluetooth 5.0 các thiết bị hoạt động tốt khi trong tầm phủ sóng có bán kính 50m. Điều này cho phép các thiết bị thông minh, đặc biệt là các thiết bị gia đình có sự kết nối chắc chắn dù ở khoảng cách xa.

Tốc độ lên đến 50 Mbps, gần đôi so với Bluetooth 4.0 chỉ có 25 Mbps.

Khả năng ít tiêu hao năng lượng khi vận hành Bluetooth. Tuy nhiên từ phiên bản Bluetooth 4.0 trở đi đã cải thiện tình hình thông qua Bluetooth Smart (LE) được cài đặt trên các thiết bị nhỏ như Smartphone, tai nghe, máy tính bảng... Và tới phiên bản 5.0, khả năng này được cải thiện lên đến 2.5 lần so với phiên bản 4.0.

Tính năng quảng cáo thông qua Bluetooth. Nếu bạn bật Bluetooth cho thiết bị di động và vô tình đi ngang qua trung tâm thương mại, cửa hàng hoặc quán ăn cũng đang sử dụng Bluetooth, nhiều khả năng bạn sẽ nhận được một đoạn quảng cáo ngắn. Chức năng này có tên là Advertising Packet, cho phép các thiết bị tự động gửi thông tin cho nhau, thông thường thì dung lượng các tin sẽ khoảng 255 bytes.

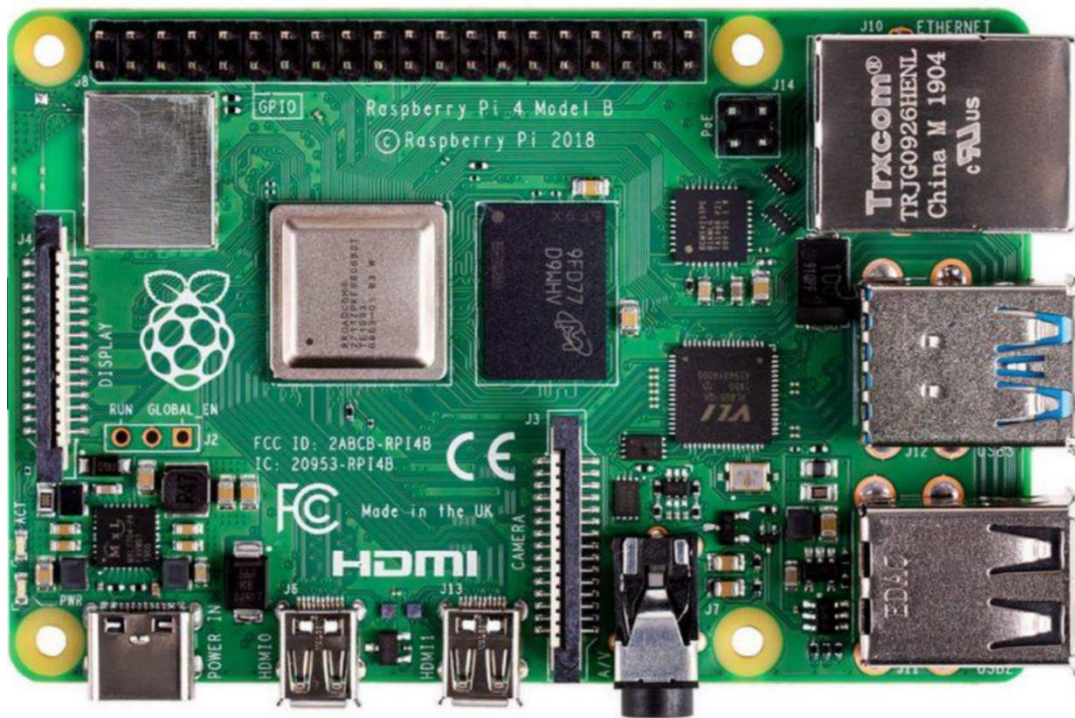
1.5. Raspberry Pi 4B

1.5.1. Tổng quan

Raspberry Pi là từ để chỉ các máy tính có một board mạch (hay còn gọi là máy tính nhúng) kích thước chỉ bằng một thẻ tín dụng, được phát triển tại Anh bởi Raspberry Pi Foundation. Raspberry Pi chủ yếu sử dụng các hệ điều hành dựa trên nhân Linux.

1.5.2. Phần cứng

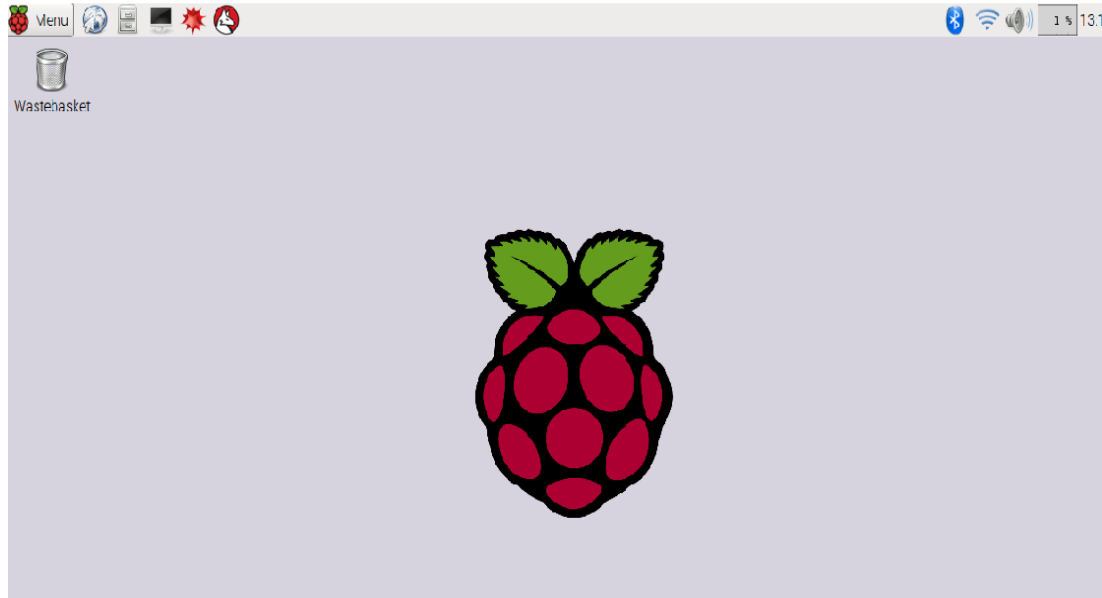
Raspberry Pi 4B được trang bị vi xử lý Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, RAM 8GB và đặc biệt hỗ trợ chuẩn Wifi 802.11n cùng với Bluetooth 5.0.



Hình 5. Raspberry Pi 4B

1.5.3. Hệ điều hành tương thích

Raspberry Pi có rất nhiều hệ điều hành hỗ trợ có nhân là Linux, trong số đó có Raspbian là hệ điều hành chính thức của Raspberry Pi Foundation. Ngoài ra có thêm một số hệ điều hành tiêu biểu như Ubuntu Mate, Window 10 IoT Core, OSMC...



Hình 6. Hệ điều hành Raspbian

1.6. API Bluetooth trên lập trình Android Studio

1.6.1. Tổng quan

- Nền tảng Android bao gồm hỗ trợ ngăn xếp mạng Bluetooth, cho phép thiết bị trao đổi dữ liệu không dây với các thiết bị Bluetooth khác. Khung ứng dụng cung cấp quyền truy cập vào chức năng Bluetooth thông qua các API Bluetooth. Các API này cho phép các ứng dụng kết nối với các thiết bị Bluetooth khác, cho phép các tính năng không dây như point to point và point to multi point.

- Sử dụng các API Bluetooth, một ứng dụng có thể thực hiện những điều sau:

- + Quét các thiết bị Bluetooth khác.
- + Truy vấn bộ điều khiển Bluetooth cục bộ cho các thiết bị Bluetooth được ghép nối.
- + Thiết lập các kênh RFCOMM.
- + Kết nối với các thiết bị khác thông qua khám phá dịch vụ.
- + Quản lý nhiều kết nối.

1.6.2. Đặc điểm

Để các thiết bị hỗ trợ Bluetooth truyền dữ liệu lẫn nhau, trước tiên chúng phải tạo một kênh giao tiếp bằng quy trình ghép nối. Một thiết bị, một thiết bị có thể khám phá, tự cung cấp cho các yêu cầu kết nối đến. Một thiết bị khác tìm thấy thiết bị có thể khám phá bằng quy trình khám phá dịch vụ. Sau khi thiết bị có thể phát hiện chấp nhận yêu cầu ghép nối, hai thiết bị sẽ hoàn tất quy trình liên kết trong đó chúng trao đổi khóa bảo mật. Thiết bị lưu vào bộ nhớ cache các phím này để sử dụng sau này. Sau khi quá trình ghép nối và liên kết hoàn tất, hai thiết bị sẽ trao đổi thông tin. Khi phiên hoàn tất, thiết bị bắt đầu yêu cầu ghép nối sẽ giải phóng kênh đã liên kết nó với thiết bị có thể khám phá. Tuy nhiên, hai thiết bị vẫn được liên kết, vì vậy chúng có thể tự động kết nối lại trong một phiên trong tương lai miễn là chúng.

1.6.3. Class Interface

android.bluetooth: Các lớp và giao diện bạn cần phải tạo để kết nối Bluetooth.

BluetoothAdapter: Đại diện cho bộ điều hợp Bluetooth cục bộ (đài Bluetooth). Đây BluetoothAdapter là điểm đầu vào cho tất cả các tương tác Bluetooth. Bằng cách sử dụng này, bạn có thể khám phá các thiết bị Bluetooth khác, truy vấn danh sách các thiết bị được liên kết (được ghép nối), khởi tạo BluetoothDevice bằng địa chỉ MAC đã biết và tạo một BluetoothSocket để nghe liên lạc từ các thiết bị khác.

BluetoothDevice: Đại diện cho một thiết bị Bluetooth từ xa. Sử dụng quyền này để yêu cầu kết nối với một thiết bị từ xa thông qua một BluetoothSocket truy vấn thông tin về thiết bị như tên, địa chỉ, lớp và trạng thái liên kết.

BluetoothSocket: Đại diện cho giao diện của ổ cắm Bluetooth (tương tự như TCP Socket). Đây là điểm kết nối cho phép ứng dụng trao đổi dữ liệu với một thiết bị Bluetooth khác bằng cách sử dụng InputStream và OutputStream.

BluetoothServerSocket: Đại diện cho một ổ cắm máy chủ mở lắng nghe các yêu cầu đến (tương tự như TCP ServerSocket). Để kết nối hai thiết bị, một thiết bị phải mở ổ cắm máy chủ với lớp này. Khi một thiết bị Bluetooth từ xa đưa ra yêu cầu kết nối với thiết bị này, thiết bị sẽ chấp nhận kết nối và sau đó trả về một kết nối đã kết nối BluetoothSocket.

BluetoothClass: Mô tả các đặc điểm và khả năng chung của thiết bị Bluetooth. Đây là một tập hợp các thuộc tính chỉ đọc để xác định các lớp và dịch vụ của thiết bị. Mặc dù thông tin này cung cấp một gợi ý hữu ích về loại thiết bị, các thuộc tính của lớp này không nhất thiết phải mô tả tất cả các cấu hình và dịch vụ Bluetooth mà thiết bị hỗ trợ.

BluetoothProfile: Giao diện đại diện cho cấu hình Bluetooth. Cấu hình Bluetooth là một đặc điểm kỹ thuật giao diện không dây cho giao tiếp dựa trên Bluetooth giữa các thiết bị. Một ví dụ là cấu hình Hands-Frees. Để biết thêm thảo luận về cấu hình, hãy xem cấu hình Bluetooth.

BluetoothHeadset: Cung cấp hỗ trợ cho tai nghe Bluetooth được sử dụng với điện thoại di động. Điều này bao gồm cả cấu hình tai nghe Bluetooth và cấu hình Hands-Free (v1.5).

BluetoothA2dp: Xác định cách truyền âm thanh chất lượng cao từ thiết bị này sang thiết bị khác qua kết nối Bluetooth bằng cấu hình phân phối âm thanh nâng cao (A2DP).

BluetoothHealth: Đại diện cho một proxy profile thiết bị sức khỏe kiểm soát dịch vụ Bluetooth.

BluetoothHealthCallback: Một lớp trừu tượng mà bạn sử dụng để triển khai các lệnh BluetoothHealth gọi lại. Bạn phải mở rộng lớp này và triển khai các phương thức gọi lại để nhận cập nhật về những thay đổi trong trạng thái đăng ký của ứng dụng và trạng thái kênh Bluetooth.

BluetoothHealthAppConfiguration: Trình bày cấu hình ứng dụng mà ứng dụng Bluetooth Health của bên thứ ba đăng ký để giao tiếp với thiết bị hỗ trợ Bluetooth từ xa.

BluetoothProfile.ServiceListener: Một giao diện thông báo cho BluetoothProfile các máy khách giao tiếp liên quá trình (IPC) khi chúng được kết nối hoặc ngắt kết nối khỏi dịch vụ nội bộ chạy một cấu hình cụ thể.

1.7. Thư viện Pybluez

1.7.1. Tổng quan

- Đặc điểm kỹ thuật của Bluetooth khá lớn và gây khó khăn cho các nhà phát triển mới làm quen. Nhưng khi kiểm tra kỹ hơn, hóa ra là mặc dù thông số kỹ thuật rất lớn, nhưng lập trình viên ứng dụng điển hình chỉ quan tâm đến một phần nhỏ của nó. Một phần quan trọng của đặc điểm kỹ thuật được dành riêng cho các nhiệm vụ cấp thấp hơn như chỉ định các tần số vô tuyến khác nhau để truyền, các giao thức thời gian và tín hiệu, và các phức tạp cần thiết để thiết lập giao tiếp. Xen kẽ với tất cả những điều này là những phần có liên quan đến nhà phát triển ứng dụng. Mặc dù Bluetooth có thuật ngữ riêng và cách mô tả các khái niệm khác nhau của nó, chúng tôi thấy rằng việc giải thích lập trình Bluetooth trong bối cảnh lập trình Internet rất dễ hiểu vì hầu hết các lập trình viên mạng đều đã quen thuộc với các kỹ thuật đó.

- Mặc dù Bluetooth đã được thiết kế từ đầu, độc lập với các giao thức Ethernet và TCP / IP, khá hợp lý khi nghĩ về lập trình Bluetooth giống như lập trình Internet. Về cơ bản, chúng có cùng nguyên tắc của một thiết bị giao tiếp và trao đổi dữ liệu với một thiết bị khác.

- Các phần khác nhau của lập trình mạng có thể được tách thành nhiều thành phần:

- + Chọn 1 thiết bị để giao tiếp.
- + Tìm ra các giao tiếp.
- + Tạo kết nối đi.
- + Chấp nhận kết nối đến.
- + Gửi dữ liệu.
- + Nhận dữ liệu.

1.7.2. Chọn giao tiếp với thiết bị

+ Mỗi chip Bluetooth từng được sản xuất đều được in bằng một địa chỉ 48-bit duy nhất trên toàn cầu, mà chúng tôi sẽ gọi là địa chỉ Bluetooth hoặc địa chỉ thiết bị. Bản chất này giống hệt với địa chỉ MAC của Ethernet, và cả hai không gian địa chỉ thực sự được quản lý bởi cùng một tổ chức - Cơ quan đăng ký IEEE. Các địa chỉ này được chỉ định tại thời điểm sản xuất và nhằm mục đích là duy nhất và giữ nguyên trạng thái tĩnh trong suốt thời gian tồn tại của chip. Nó thuận tiện đóng vai trò là đơn vị định địa chỉ cơ bản trong tất cả các chương trình Bluetooth.

+ Để một thiết bị Bluetooth giao tiếp với thiết bị khác, thiết bị đó phải có một số cách xác định địa chỉ Bluetooth của thiết bị kia. Địa chỉ này được sử dụng ở tất cả các lớp của quá trình giao tiếp Bluetooth, từ các giao thức vô tuyến cấp thấp đến các giao thức ứng dụng cấp cao hơn. Ngược lại, các thiết bị mạng TCP / IP sử dụng Ethernet làm lớp liên kết dữ liệu của chúng sẽ loại bỏ địa chỉ MAC 48 bit ở các lớp cao hơn của quá trình truyền thông và chuyển sang sử dụng địa chỉ IP. Tuy nhiên, nguyên tắc vẫn giống nhau, trong đó địa chỉ nhận dạng duy nhất của thiết bị đích phải được biết để giao tiếp với nó.

+ Trong cả hai trường hợp, chương trình client thường sẽ không có kiến thức trước về các địa chỉ đích này. Trong lập trình Internet, người dùng thường sẽ cung cấp tên máy chủ, chẳng hạn như ctu.edu.vn , tên máy khách phải dịch sang địa chỉ IP vật lý bằng Hệ thống tên miền (DNS). Trong Bluetooth, người dùng thường sẽ cung cấp một số tên thân thiện với người dùng, chẳng hạn như “HuyGao” và ứng dụng khách chuyển tên này thành địa chỉ số bằng cách tìm kiếm các thiết bị Bluetooth gần đó.

1.7.3. Giao thức kết nối

- Khi ứng dụng khách của chúng tôi đã xác định được địa chỉ của máy chủ mà nó muốn kết nối, nó phải xác định giao thức truyền tải nào sẽ sử dụng. Phần này mô tả các giao thức truyền tải Bluetooth có bản chất gần nhất với các giao thức Internet được sử dụng phổ biến nhất. Cũng cần xem xét cách lập trình viên có thể chọn giao thức nào để sử dụng dựa trên các yêu cầu của ứng dụng.

- Cả lập trình Bluetooth và Internet đều liên quan đến việc sử dụng nhiều giao thức truyền tải khác nhau, một số giao thức được xếp chồng lên nhau. Trong TCP / IP, nhiều ứng dụng sử dụng TCP hoặc UDP, cả hai ứng dụng này đều dựa vào IP như một phương tiện truyền tải cơ bản. TCP cung cấp một phương pháp hướng kết nối để gửi dữ liệu trong các luồng một cách đáng tin cậy và UDP cung cấp một lớp bọc mỏng xung quanh IP để gửi các gói dữ liệu riêng lẻ có độ dài tối đa cố định một cách không đáng tin cậy. Ngoài ra còn có các giao thức như RTP cho các ứng dụng như liên lạc thoại và video có yêu cầu nghiêm ngặt về thời gian và độ trễ.

- Mặc dù Bluetooth không có các giao thức tương đương chính xác, nhưng nó cung cấp các giao thức thường có thể được sử dụng trong các ngữ cảnh giống như một số giao thức Internet.

1.7.3.1. Giao thức RFCOMM + TCP

- Giao thức RFCOMM cung cấp dịch vụ và đảm bảo độ tin cậy gần giống như TCP. Mặc dù thông số kỹ thuật nói rõ ràng rằng nó được thiết kế để mô phỏng các cổng nối tiếp RS-232 (để giúp các nhà sản xuất thêm khả năng Bluetooth vào các thiết bị cổng nối tiếp hiện có của họ dễ dàng hơn), nó khá đơn giản để sử dụng nó trong nhiều trường hợp tương tự như TCP.

- Nói chung, các ứng dụng sử dụng TCP quan tâm đến việc có một kết nối point to point mà qua đó chúng có thể trao đổi các luồng dữ liệu một cách đáng tin cậy. Nếu một phần của dữ liệu đó không thể được phân phối trong một giới hạn thời gian cố định, thì kết nối sẽ bị chấm dứt và sẽ xảy ra lỗi. Cùng với các thuộc tính mô phỏng cổng nối tiếp khác nhau mà phần lớn không liên quan đến các nhà lập trình mạng, RFCOMM cung cấp các thuộc tính chính tương tự của TCP.

1.7.3.2. Service Discovery Protocol

- Chúng ta tìm ra cách giao tiếp với một máy từ xa, khi đã biết địa chỉ số và giao thức truyền tải, là chọn số cổng. Hầu hết tất cả các giao thức truyền tải Internet được sử dụng phổ biến được thiết kế với khái niệm số cổng, do đó nhiều ứng dụng trên cùng một máy chủ có thể đồng thời sử dụng một giao thức truyền tải. Bluetooth không phải là ngoại lệ, nhưng sử dụng thuật ngữ hơi khác một chút. Trong L2CAP, các cổng được gọi là Bộ ghép kênh dịch vụ giao thức và có thể nhận các giá trị được đánh số lẻ từ 1 đến 32767. Trong RFCOMM, các kênh 1-30 có sẵn để sử dụng. Bên cạnh những khác biệt này, cả bộ ghép kênh và kênh dịch vụ giao thức đều phục vụ cùng một mục đích mà các cổng thực hiện trong TCP / IP. L2CAP, không giống như RFCOMM, có một loạt các số cổng dành riêng (1-1023) không được sử dụng cho các ứng dụng và giao thức tùy chỉnh. Thông tin này được tóm tắt trong “Hình 1.7.1”. Thông qua phần còn lại của tài liệu này, cổng từ được sử dụng thay cho bộ ghép kênh và kênh dịch vụ giao thức để rõ ràng hơn.

protocol	terminology	reserved/well-known ports	dynamically assigned ports
TCP	port	1-1024	1025-65535
UDP	port	1-1024	1025-65535
RFCOMM	channel	none	1-30
L2CAP	PSM	odd numbered 1-4095	odd numbered 4097 - 32765

Hình 7. So sánh các Protocol của thư viện Pybluez

1.7.3.3. Thư viện Pybluez

lookup_names: Đặt thành True nếu bạn muốn tra cứu thân thiết với người dùng.

lookup_class: Cố gắng tìm kiếm lớp của thiết bị được phát hiện.

server_name: Đặt thành tên của dịch vụ nào mà bạn muốn.

inquiry_complete(): Được gọi khi một yêu cầu bắt đầu bởi **find_devices** đã hoàn thành.

BluetoothSocket(): Khai báo server Bluetooth Socket là giao thức nào.

+ RFCOMM.

+ L2CAP.

server_sock.bind(): Khai báo port.

server_sock.listen(): Lắng nghe trên channel máy.

client_sock.recv(): Giữ dữ liệu với client.

client_sock.send(): Nhận dữ liệu từ client.

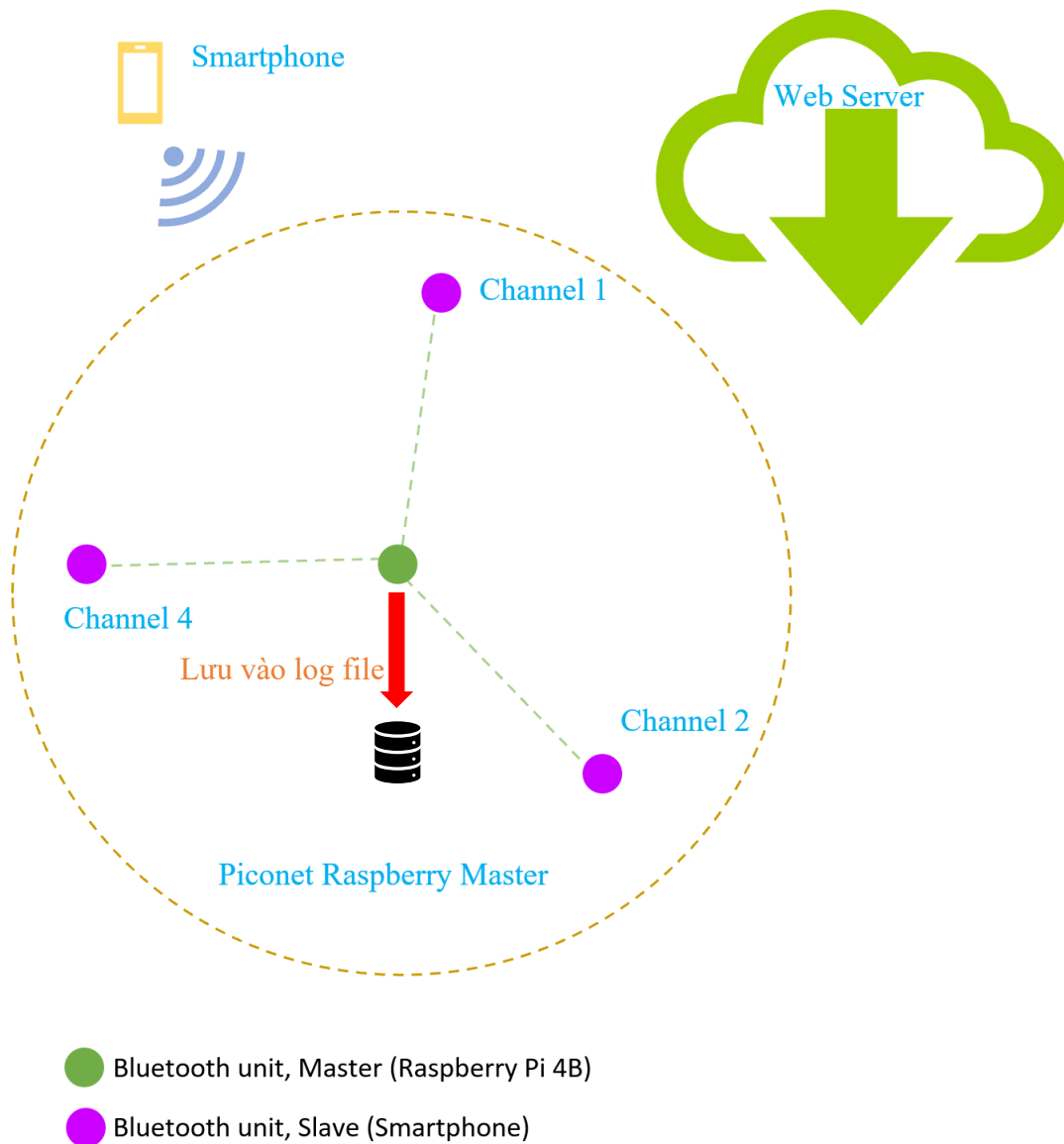
client_sock.close(): Đóng kết nối của client.

server_sock.close(): Đóng kết nối của server.

Chương 2. Thiết Kế Giải Pháp

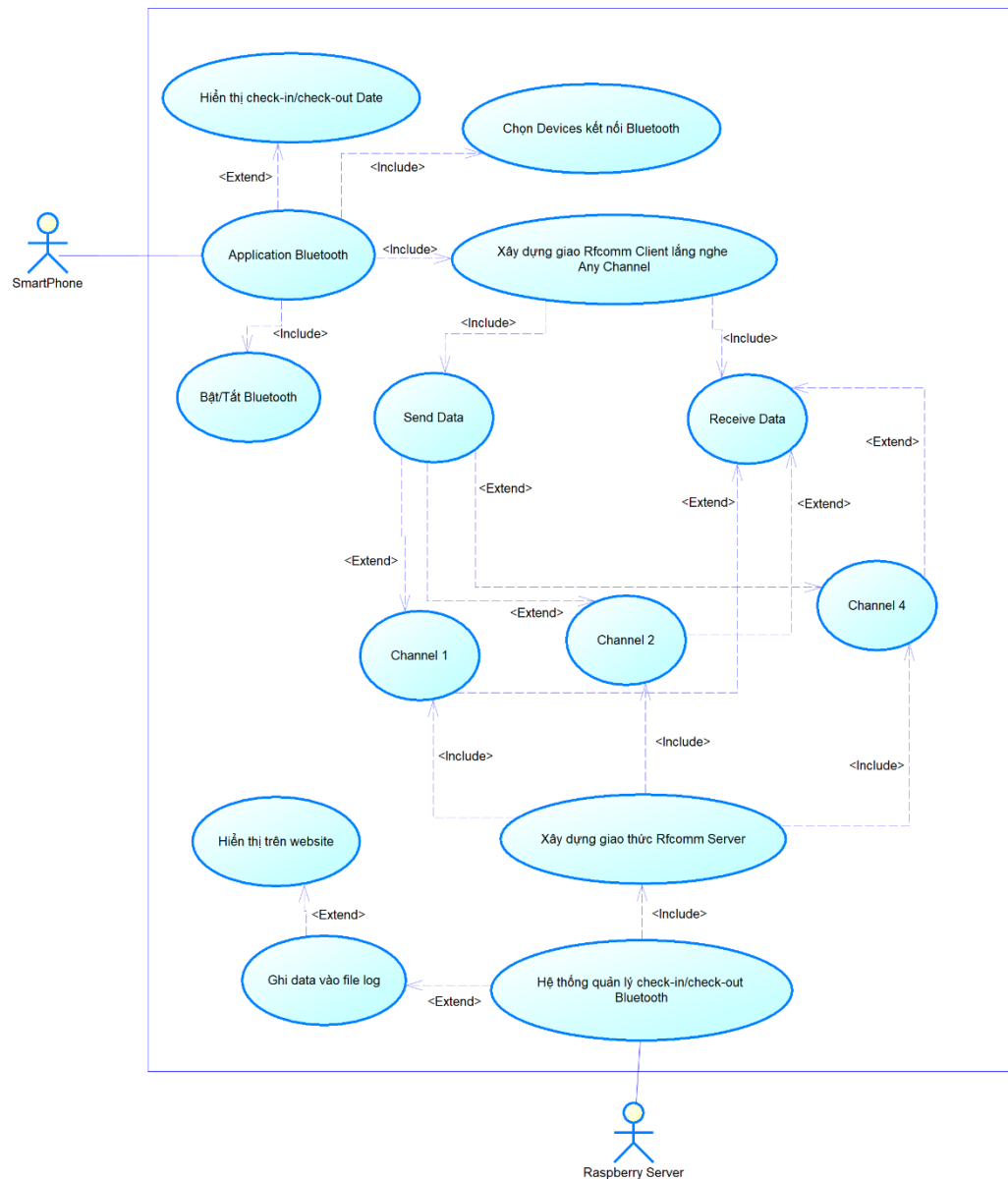
1. Kiến trúc hệ thống:

1.1. Mô hình tổng thể các thành phần trong hệ thống



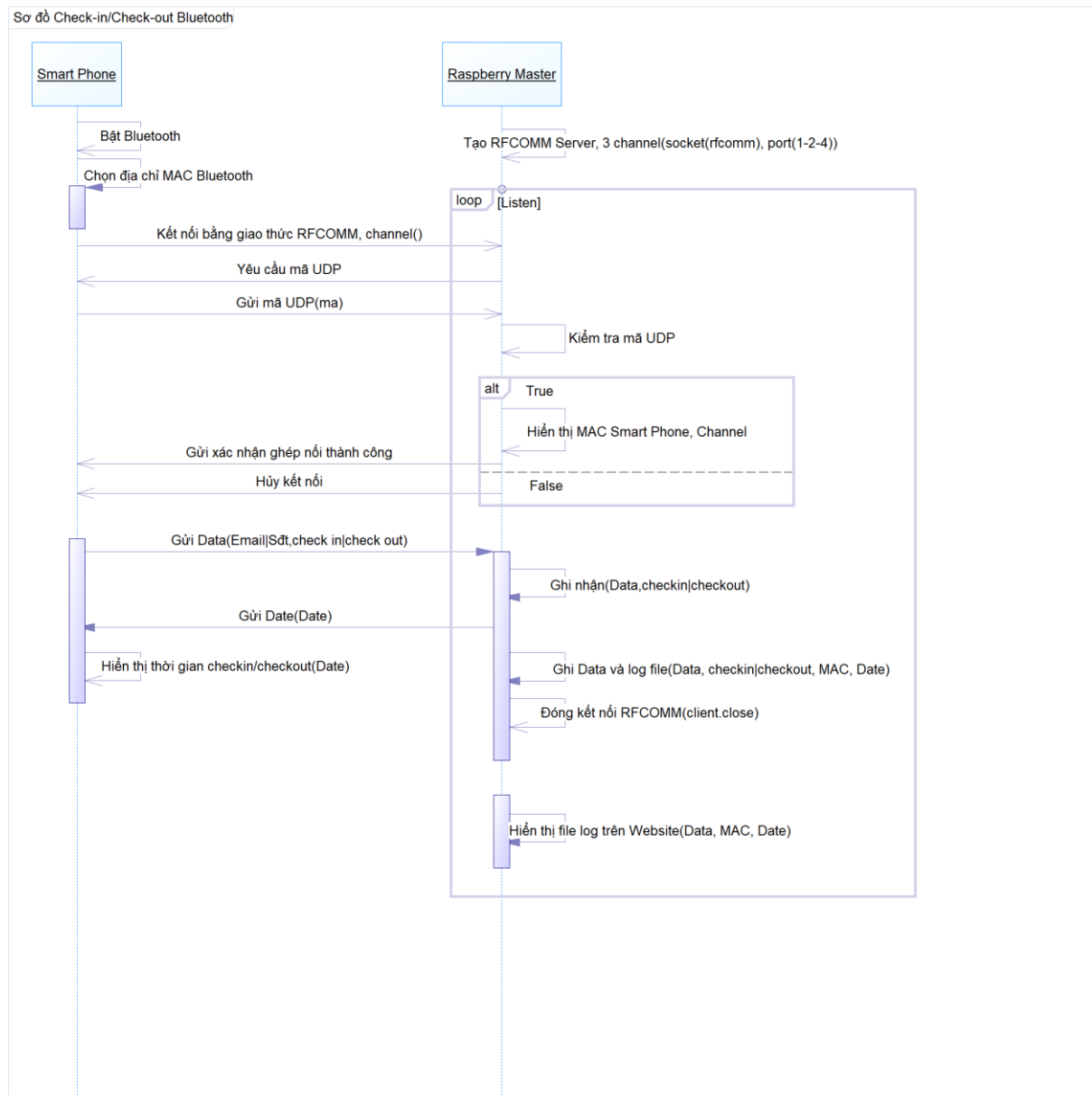
Hình 8. Mô hình Bluetooth Check_in

1.2. Mô hình Use case Hệ thống thu thập dữ liệu check in dựa trên Bluetooth



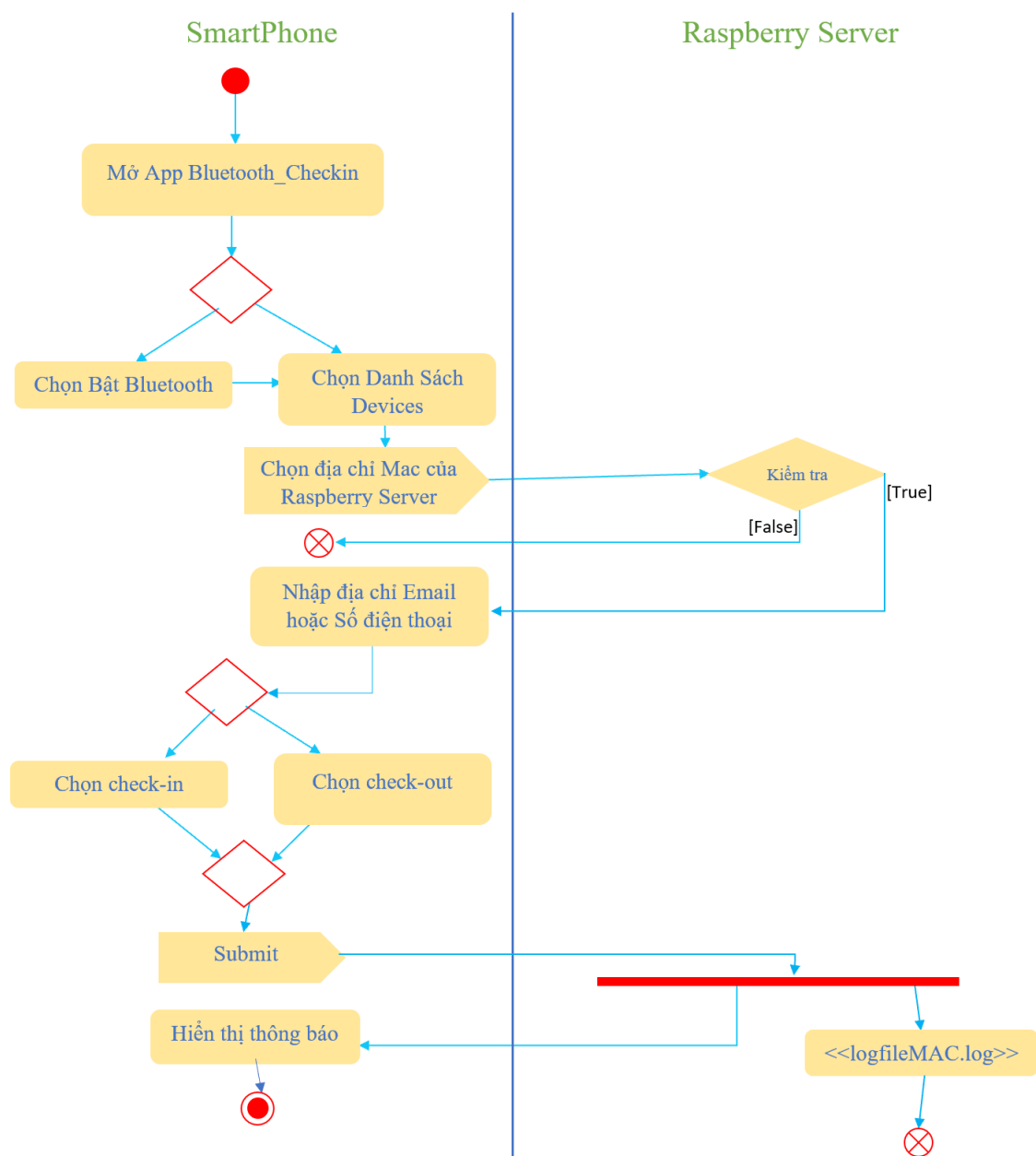
Hình 9. Mô hình Use case

1.3. Mô hình Sequence Hệ thống thu thập dữ liệu check in dựa trên Bluetooth



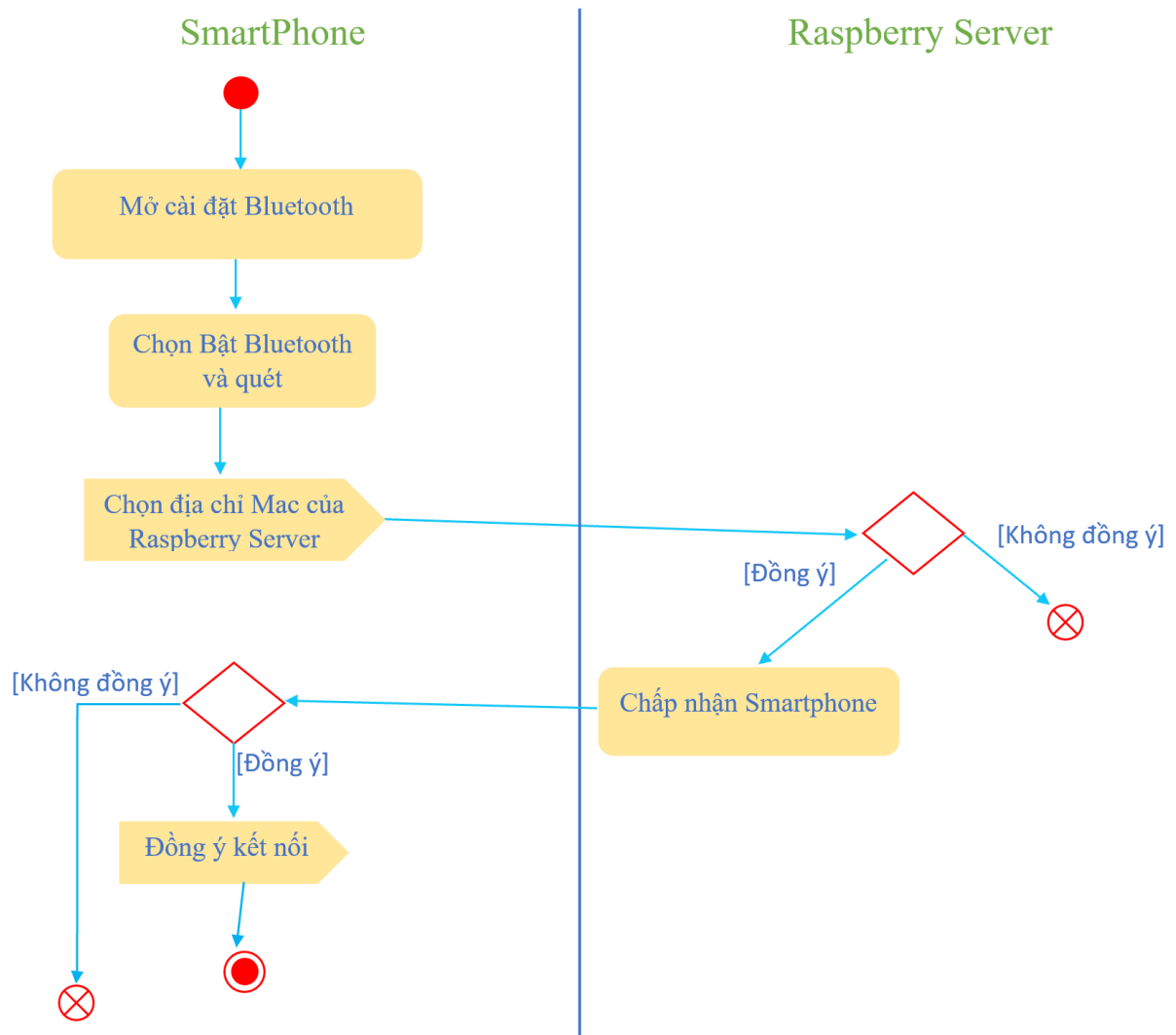
Hình 10. Mô hình Sequence

1.4. Data Flow Diagram Hệ thống thu thập dữ liệu check in dựa trên Bluetooth



Hình 11. Data Flow Diagram

1.5. Data Flow Diagram đăng ký active bluetooth lần đầu



Hình 12. Data flow diagram đăng ký bluetooth lần đầu

2. Thiết kế giải thuật xử lý

2.1. Trên Raspberry Pi 4B

- Bắt đầu xây dựng phần mềm thu thập dữ liệu check in dựa trên Bluetooth, ta tạo một project với tên **HethongCheckin-Bluetooth-sv.py**.

- Khai báo thư viện gồm:

```
from datetime import datetime
import threading
from bluetooth import *
```

- Khai báo chương trình con:

```
+ def blue1()
```

- Cho **try** vào để bắt lỗi ngoại trừ các lỗi không mong muốn

- Khai báo hàm **datetime** để ghi nhận thời gian Smartphone check in:

```
now = datetime.now()
dt_string = now.strftime("%d/%m/%Y %H:%M:%S")
date_string = "Xin chào! Bạn đã đăng ký thành công vào lúc: " + dt_string
```

- Khai báo hàm lớp BluetoothSocket và chỉ giao thức là RFCOMM trên port 1 tiếp đó là lắng nghe chờ Smartphone kết nối:

```
server_sock=BluetoothSocket(RFCOMM)
port = 1
server_sock.bind(("",port))
server_sock.listen(1)
```

- Khai báo quản bá dịch vụ (Service Discovery Protocol) để bắt buộc phía client trên phần mềm check in của Smartphone phải xuất trình mã định danh (uuid) thì mới cho xác thực kết nối. Trong mô hình này thì chuỗi định danh là duy nhất và có độ dài 128 bit:

```
uuid = "94f39d29-7d6d-437d-973b-fba39e49d4ee"
```

- Tiếp theo khai báo thông tin dịch vụ quản bá của Raspberry Server, bao gồm tên:

```
advertise_service( server_sock, "Rpi4", service_id = uuid)
```

- Chấp nhận kết nối và in thông tin của client vừa kết nối(Smartphone):

```
client_sock, client_info = server_sock.accept()
```

```
print("Đã kết nối thành công tới client ", client_info)
```

- Gửi dữ liệu đến Smartphone thông báo đã check in thành công, ngay sau khi thiết lập kết nối. Cho kích thước dữ liệu là 1024 bytes:

```
data = client_sock.recv(1024)
```

- Tiếp theo nhận dữ liệu check in từ Smartphone gửi và lưu dữ liệu:

```
if len(data) == 0: break
```

```
print("Client gửi~: %s" % data)
```

```
datadecode = data.decode('ASCII')
```

```
client_sock.send(date_string)
```

```
f.open("/var/www/html/logfileMAC.log","a")
```

```
f.write(datadecode + " -- " + tam1 + " -- " + dt_string + "\n")
```

```
f.close()
```

- Đóng kết nối Bluetooth:

```
client_sock.close()
```

```
server_sock.close()
```

- Khởi tạo chạy đa luồng:

```
b1 = threading.Thread(target = blue1)
```

- Tạo vòng lặp while để cứ sau 1 giây thì các port được làm mới:

```
while True:
```

```
    if not (b1.isAlive())
```

```
        b1new = threading.Thread(target = blue1)
```

```
        b1new.start()
```

```
        time.sleep(1)
```

- Tương tự với chương trình con blue1() với port 1 thì ta tạo thêm 2 channel (port 2, port 4) để tăng số lượng chịu tải.

2.2. Trên Android Studio Lập trình giao diện mức thấp trên MainActivity.java

- Khai báo thư viện:

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import java.util.Set;
import java.util.ArrayList;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ArrayAdapter;
import android.content.Intent;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
```

- Khai báo API bluetooth:

```
private BluetoothAdapter BTAdapter;
private Set<BluetoothDevice>pairedDevices;
```

- Mở Bluetooth trên Smartphone:

```

if (!BTAdapter.isEnabled()) {
    Intent turnOn = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    //Thực hiện việc mở bluetooth. Yêu cầu bật bluetooth
    startActivityForResult(turnOn, 0);
    Toast.makeText(getApplicationContext(), "Mở Device
Bluetooth", Toast.LENGTH_SHORT).show();
} else {
    Toast.makeText(getApplicationContext(), "Thiết bị đã mở Device
Bluetooth", Toast.LENGTH_SHORT).show();
}
scanbt.setVisibility(View.VISIBLE);
lv.setVisibility(View.VISIBLE);
}

```

- Tắt Bluetooth trên Smartphone:

```

public void off(View v){
    BTAdapter.disable();
    Toast.makeText(getApplicationContext(), "Tắt Device Bluetooth"
, Toast.LENGTH_SHORT).show();
    scanbt.setVisibility(View.INVISIBLE);
    lv.setVisibility(View.GONE);
}

```

- Hiện thị các địa chỉ MAC Bluetooth trên màn hình:

```

public void deviceList(View v){
    ArrayList deviceList = new ArrayList();
    pairedDevices = BTAdapter.getBondedDevices();

    if (pairedDevices.size() < 1) {
        Toast.makeText(getApplicationContext(), "Không tìm thấy Device Bluetooth
để kết nối...", Toast.LENGTH_SHORT).show();
    } else {
        for (BluetoothDevice bt : pairedDevices) deviceList.add(bt.getName() + "
" + bt.getAddress());
        Toast.makeText(getApplicationContext(), "Hiện thị các Devices đã được kết
nối...", Toast.LENGTH_SHORT).show();
        final ArrayAdapter adapter = new ArrayAdapter(this,
android.R.layout.simple_list_item_1, deviceList);
        lv.setAdapter(adapter);
        lv.setOnItemClickListener(myItemClickListener);
    }
}

```

2.3. Lập trình giao diện mức thấp trên CommsActivity.java

- Khai báo thư viện cần thiết:

```
import androidx.appcompat.app.AppCompatActivity;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;
```

- Khai báo API Bluetooth:

```
public BluetoothAdapter BTAdapter = BluetoothAdapter.getDefaultAdapter();
```

- Kết nối Giao thức RFCOMM với Raspberry Master và gửi dịch vụ quản bá(SDP). Nếu đúng thì sẽ kết nối thành công nếu sai thì báo lỗi:

```
private ConnectThread(BluetoothDevice device) throws IOException {
    BluetoothSocket tmp = null;
    mmDevice = device;
    try {
        UUID uuid = UUID.fromString("94f39d29-7d6d-437d-973b-fba39e49d4ee");
        tmp = mmDevice.createRfcommSocketToServiceRecord(uuid);
    } catch (IOException e) {
        Log.e(TAG, "Socket's create() khởi tạo không thành công!", e);
    }
    mmSocket = tmp;
    BTAdapter.cancelDiscovery();
    try {
        mmSocket.connect();
    } catch (IOException connectException) {
        Log.v(TAG, "Lỗi kết nối!");
        try {
            mmSocket.close();
        } catch (IOException closeException) {
        }
    }
    send();
}
```

- Gửi khai báo check in với Raspberry Master mà người dùng nhập:

```
public void send() throws IOException {
    String msg = editTextName.getText().toString() + "#" + radioButton.getText();
    OutputStream mmOutputStream = mmSocket.getOutputStream();
    mmOutputStream.write(msg.getBytes());
    receive();
}
```

- Hiển thị nội dung ra màn hình khi đã check in thành công:

```
public void receive() throws IOException {  
    InputStream mmInputStream = mmSocket.getInputStream();  
    //tao mảng byte 1024 bytes  
    byte[] buffer = new byte[1024];  
    int bytes;  
  
    try {  
        bytes = mmInputStream.read(buffer);  
        String readMessage = new String(buffer, 0, bytes);  
        Log.d(TAG, "Nội dung nhận: " + readMessage);  
        TextView content = (TextView) findViewById(R.id.Data1);  
        //Hiển thị nội dung ra màn hình  
        content.setText(readMessage );  
        mmSocket.close();  
    } catch (IOException e) {  
        Log.e(TAG, "Lỗi nhận nội dung!");  
        return;  
    }  
}
```

Chương 3. Cài Đặt Giải Pháp

3.1. Cài đặt và quản trị hệ điều hành Raspbian trên Raspberry Pi

1. Chuẩn bị thẻ nhớ microSD có dung lượng tối thiểu 8GB và đã được định dạng.
2. Download hệ điều hành Raspbian về máy tính bằng đường link: https://downloads.raspberrypi.org/imager/imager_1.4.exe.
3. Download phần mềm balenaEtcher giải nén hệ điều hành Raspbian vào thẻ microSD.
4. Mở terminal gõ lệnh: `sudo raspi -config`, lúc này màn hình hiển thị hộp thoại Raspberry Software Configuration Tool -> Interfacing Options -> SSH để có thể mở port 22 để cho phép truy cập từ xa vào Raspberry Pi bằng dòng lệnh và mở port 5900 để truy cập từ xa bằng giao diện GUI.
5. Truy cập từ xa ssh từ máy trạm: `ssh pi@IP` của Raspberry và nhập mật khẩu là raspberry (mặc định).

3.2. Thiết lập môi trường phát triển ứng dụng Pybluez

1. Khi kết nối được Raspberry Pi bằng SSH, tiếp theo sử dụng lệnh: **`sudo apt update`** để cập nhật lại hệ điều hành.
2. Cài đặt biến môi trường Pybluez trên Raspberry Pi: **`sudo apt-get install pybluez libbluetooth-dev -y`**
3. Kiểm tra xem info Bluetooth: **`sudo service bluetooth status`**

3.3. Cấp quyền cho ứng dụng Bluetooth_Checkin trên Android Studio

1. Vào Project dự án của mình `/app/manifests/AndroidManifest.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.bluetooth_checkin">

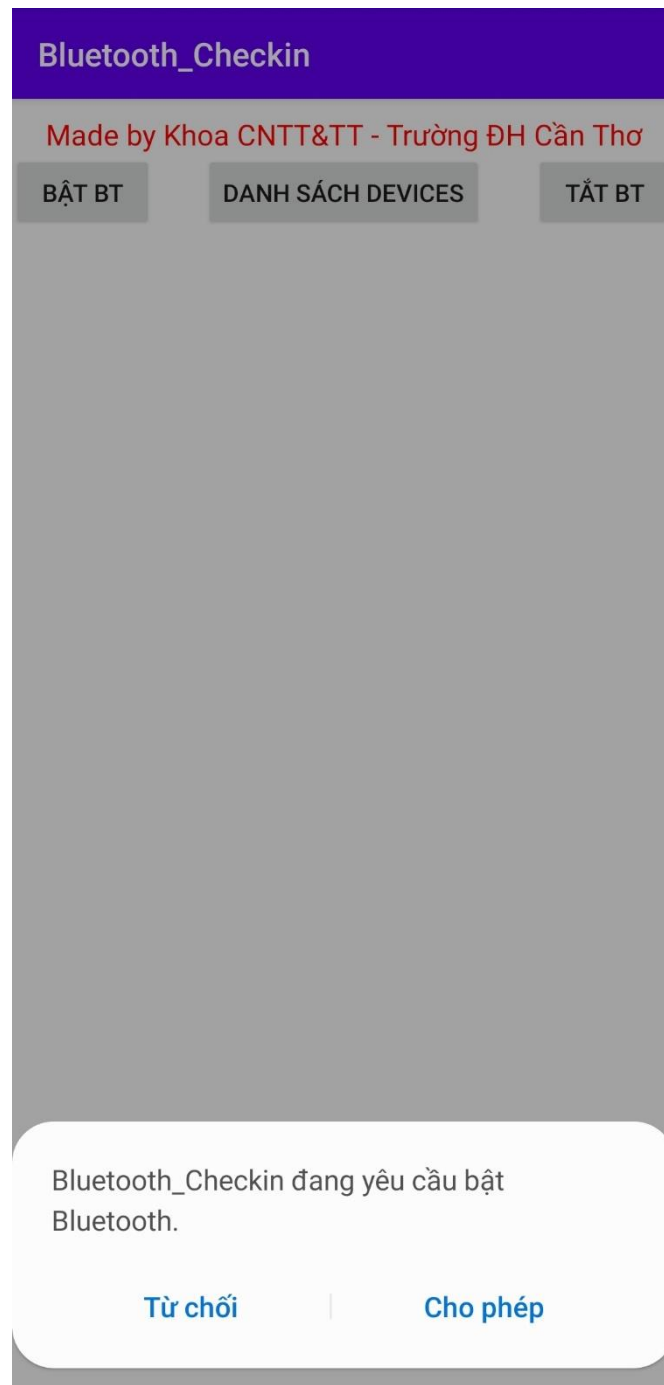
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".CommsActivity"></activity>
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

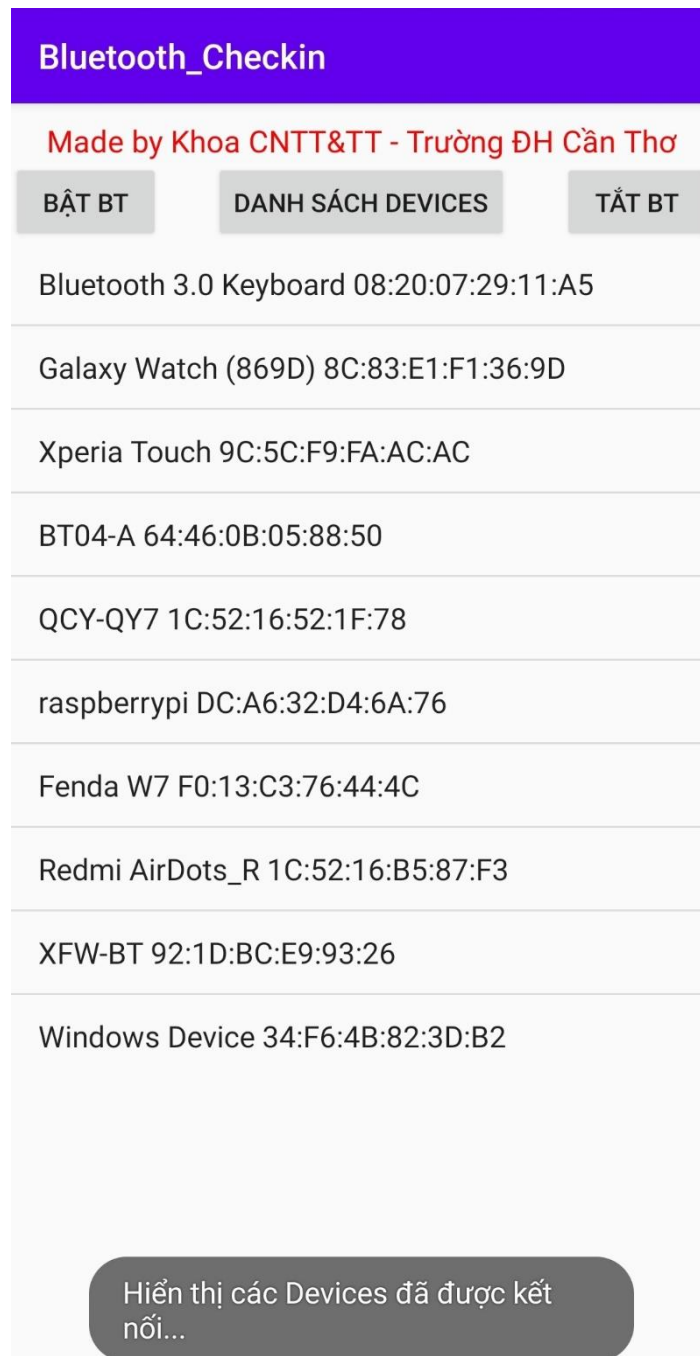
3.4. Khởi chạy ứng dụng Bluetooth_checkin trên Smartphone

1. Mở ứng dụng Bluetooth_checkin và bật nút “Bật BT”.



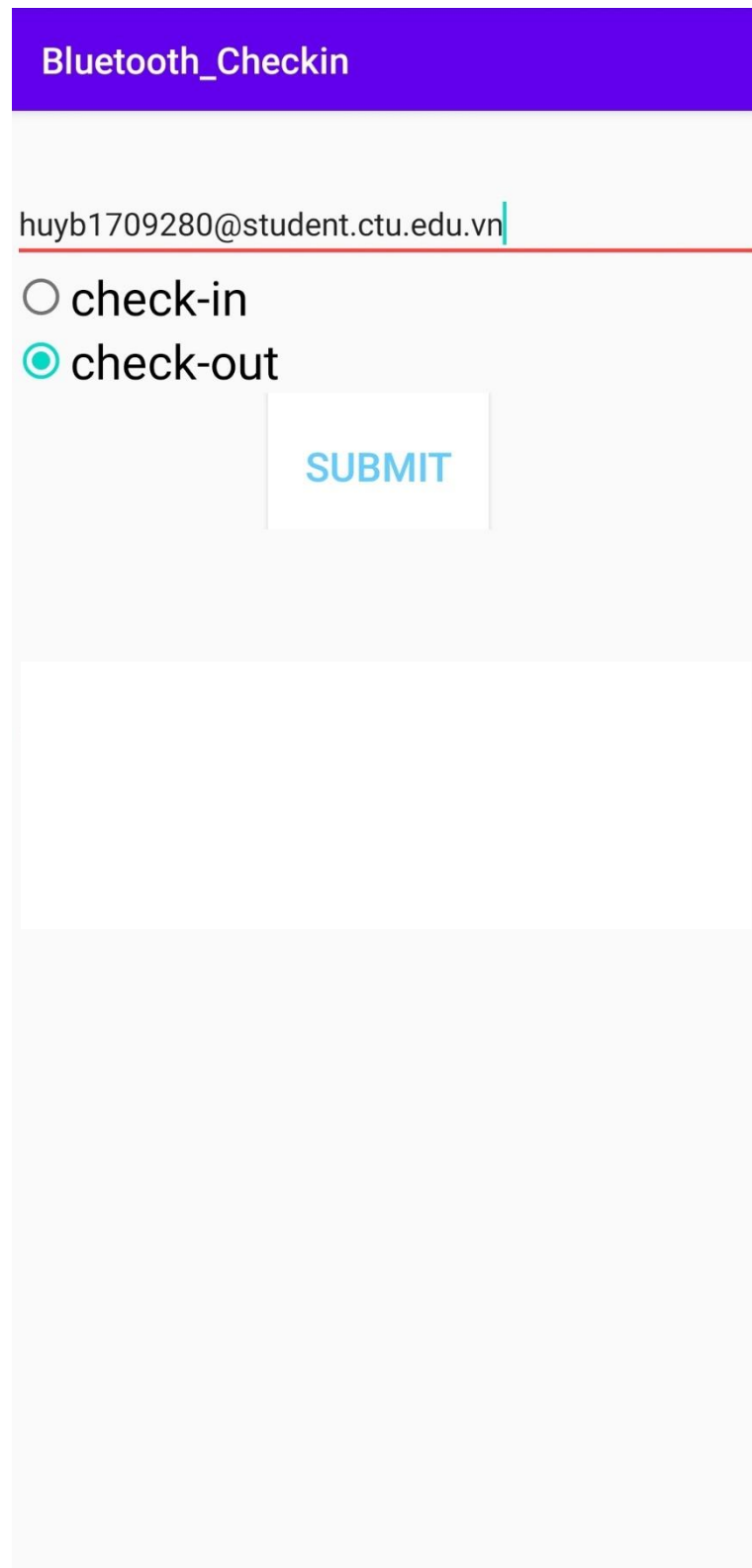
Hình 13. Giao diện chính của ứng dụng Bluetooth_Checkin

2. Bấm vào “DANH SÁCH DEVICES” và tìm địa chỉ MAC Bluetooth của Hệ thống thu thập dữ liệu check in.



Hình 14. Danh sách device bluetooth

3. Nhập vào thông tin để khai báo check in hoặc check out.

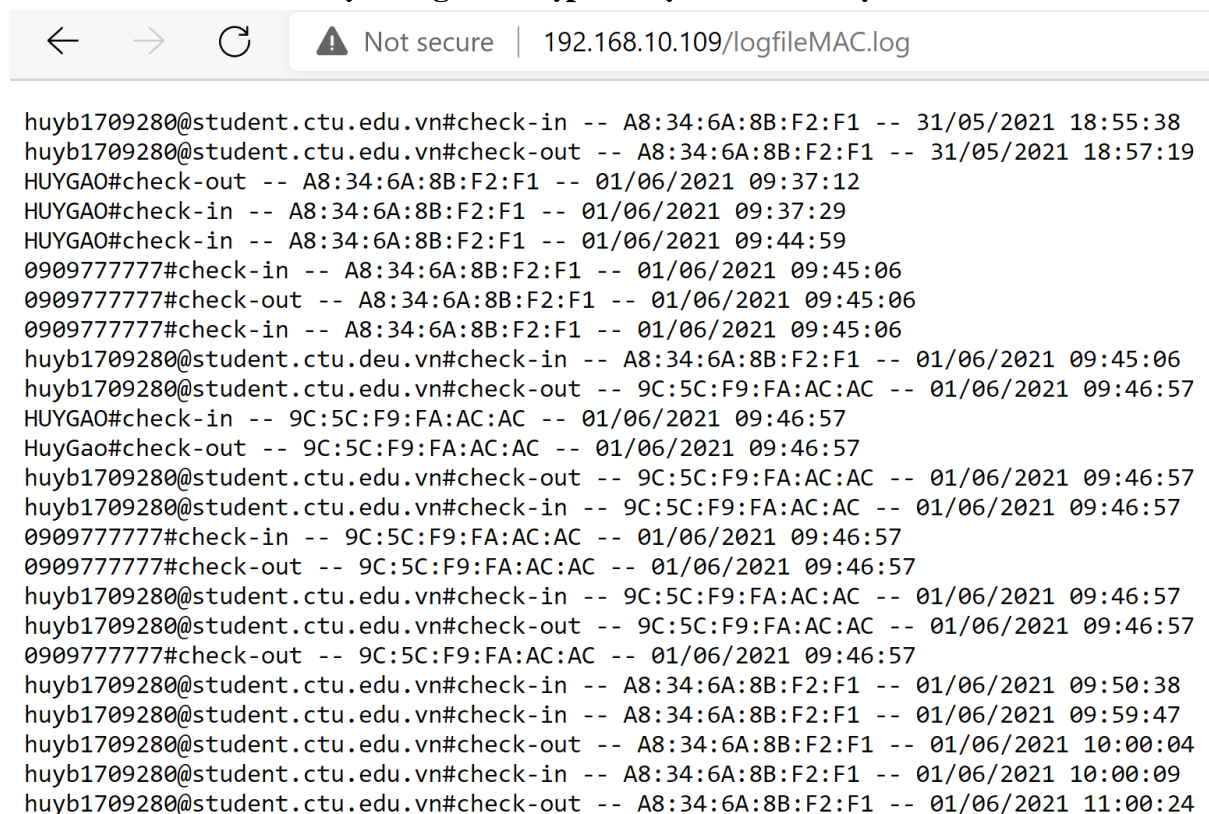


The screenshot shows a web form titled "Bluetooth_Checkin" with a purple header. Below the header is a text input field containing the email "huyb1709280@student.ctu.edu.vn". Underneath the input field are two radio button options: "check-in" and "check-out". The "check-out" option is selected, indicated by a teal dot. To the right of these options is a light blue "SUBMIT" button. Below the form fields is a large, empty white rectangular area, and the bottom of the page is a light gray section.

Hình 15. Giao diện người dùng nhập dữ liệu check in – check out

3.5. Kiểm tra kết quả

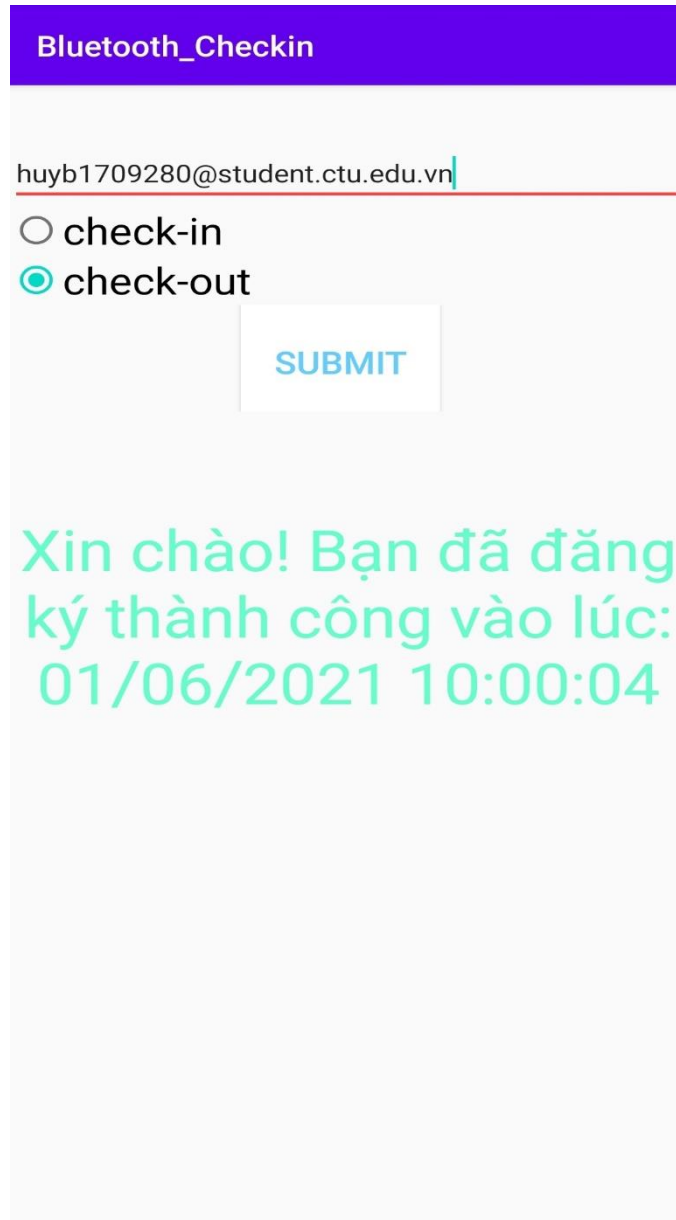
3.5.1 Trên Hệ thống thu thập dữ liệu check in dựa trên Bluetooth



Hình 16. Dữ liệu file log ghi nhận được

Kết quả cho ta thấy đã nhận được tính hiệu từ Smartphone và lưu thành công gồm thông tin người dùng nhập, check in hay check out. Địa chỉ MAC Bluetooth để xác thực người dùng và trên thế giới chỉ có đúng 1 địa chỉ này, tiếp theo là ngày giờ ra vào.

3.5.2. Trên ứng dụng Bluetooth_checkin của Smartphone



The screenshot shows the 'Bluetooth_Checkin' app interface. At the top, there is a purple header with the text 'Bluetooth_Checkin'. Below the header, the email address 'huyb1709280@student.ctu.edu.vn' is displayed. Underneath the email, there are two radio button options: 'check-in' (which is unselected) and 'check-out' (which is selected). A 'SUBMIT' button is located to the right of these options. At the bottom of the screen, a large green message reads: 'Xin chào! Bạn đã đăng ký thành công vào lúc: 01/06/2021 10:00:04'.

Hình 17. Thông báo check in thành công

Kết quả cho ta thấy Smartphone đã kết nối thành công với hệ thống check in Bluetooth và nhận về tin nhắn do Raspberry Master gửi. Ứng dụng hoạt động tốt.

Chương 4. Kiểm Thử Và Đánh Giá

1. Kiểm thử

Hệ thống thu thập dữ liệu check in dựa trên Bluetooth có kết quả chính xác tuyệt đối về ghi nhận và trả dữ liệu về cho Smartphone, không xảy ra hiện tượng lỗi và độ trễ. Đạt được mục tiêu đề ra ghi nhận thông tin người check in và check out.

2. Đánh giá

Mục tiêu thu thập dữ liệu ra vào của người dùng thành công, ứng dụng trên Smartphone hoạt động ổn định, hệ thống thu thập dữ liệu đã ghi nhận dữ liệu thành công.

Có thể biến Raspberry Pi thành một hệ thống thu thập dữ liệu người ra vào trong cty, trường học, bệnh viện hoặc nơi công cộng.... nhằm biết được lịch sử người đó đi lại để biết những người có tiếp xúc với bệnh nhân nhiễm bệnh hoặc nghi là nhiễm Covid-19 không? Và tạo ra sự tiện lợi cho người dùng và an toàn những người trực ban phải tiếp xúc vs những người nghi là nhiễm bệnh.

PHẦN III. KẾT LUẬN

1. Kết quả đạt được

Hệ thống thu thập dữ liệu check in dựa trên Bluetooth đã cho kết quả chính xác, chu trình của mô hình khép kín hoạt động liên tục, dữ liệu được ghi nhận trên file log khi có người đi qua vùng phủ sóng của Raspberry Master.

Ghi nhận lịch sử check in – check out theo thời gian thực, giúp người dùng dễ giám sát và trực quan khi xem.

Giúp giảm thiểu nguy cơ lây bệnh cho mọi người và cộng đồng.

2. Hướng phát triển

Tìm hiểu sâu hơn về lập trình Bluetooth bằng ngôn ngữ Python trên Raspberry Pi cũng như các ngôn ngữ lập trình khác.

Mở rộng mô hình xây dựng hệ thống thu thập dữ liệu hoàn chỉnh hơn.

Tìm hiểu và xây dựng hệ thống thu thập dữ liệu người dùng thông minh hơn.

Xây dựng cơ sở dữ liệu lưu trữ trên đám mây giúp dễ dàng mở rộng và giảm bớt xử lý cho server.

TÀI LIỆU THAM KHẢO

- [1] <https://pybluez.readthedocs.io/en/latest/index.html>
- [2] <https://developer.android.com/guide/topics/connectivity/bluetooth/>