

Số: ~~3071~~/QĐ-UBND

Thừa Thiên Huế, ngày ~~26~~ tháng ~~11~~ năm 2021

**QUYẾT ĐỊNH**

**Ban hành Hướng dẫn kỹ thuật tích hợp ứng dụng vào ứng dụng  
dịch vụ đô thị thông minh Hue-S**

**ỦY BAN NHÂN DÂN TỈNH THỪA THIÊN HUẾ**

*Căn cứ Luật Tổ chức chính quyền địa phương ngày 19 tháng 6 năm 2015;  
Căn cứ Luật sửa đổi, bổ sung một số điều của Luật Tổ chức Chính phủ và Luật  
Tổ chức chính quyền địa phương ngày 22 tháng 11 năm 2019;*

*Căn cứ Luật Giao dịch điện tử ngày 29 tháng 11 năm 2005;*

*Căn cứ Luật Công nghệ thông tin ngày 29 tháng 6 năm 2006;*

*Căn cứ Luật An toàn thông tin mạng ngày 19 tháng 11 năm 2015;*

*Căn cứ Nghị định số 64/2007/NĐ-CP ngày 10 tháng 4 năm 2007 của  
Chính phủ ứng dụng Công nghệ thông tin trong hoạt động của cơ quan nhà  
nước;*

*Căn cứ Nghị định số 47/2020/NĐ-CP ngày 09 tháng 4 năm 2020 của  
Chính phủ về quản lý, kết nối và chia sẻ dữ liệu số của cơ quan Nhà nước;*

*Căn cứ Nghị quyết số 12-NQ/TU ngày 12 tháng 11 năm 2021 của Hội  
nghị lần thứ năm Ban chấp hành Đảng bộ tỉnh khóa XVI về chuyển đổi số tỉnh  
Thừa Thiên Huế đến năm 2025, định hướng đến năm 2030;*

*Căn cứ Quyết định số 62/2021/QĐ-UBND ngày 04 tháng 10 năm 2021  
của UBND tỉnh Thừa Thiên Huế về việc Ban hành Quy định Quản lý, vận hành  
và tích hợp ứng dụng Hue-S;*

*Theo đề nghị của Giám đốc Sở Thông tin và Truyền thông tỉnh tại Tờ  
trình số: 2462/TTr-STTTT ngày 01 tháng 11 năm 2021.*

**QUYẾT ĐỊNH:**

**Điều 1.** Ban hành kèm theo Quyết định này Hướng dẫn kỹ thuật tích hợp  
ứng dụng vào ứng dụng dịch vụ đô thị thông minh Hue-S .

**Điều 2. Hiệu lực thi hành**

Quyết định này có hiệu lực thi hành kể từ ngày ký.


### **Điều 3. Trách nhiệm thi hành**

Chánh Văn phòng Ủy ban nhân dân tỉnh; Giám đốc Sở Thông tin và Truyền thông; Thủ trưởng các Sở, ban, ngành, đoàn thể cấp tỉnh; Chủ tịch Ủy ban nhân dân các huyện, thị xã, thành phố Huế; Chủ tịch Ủy ban nhân dân các xã, phường, thị trấn; Thủ trưởng các cơ quan, đơn vị; tổ chức, cá nhân có liên quan chịu trách nhiệm thi hành Quyết định này./.

#### **Nơi nhận:**

- Như Điều 3;
- Thường vụ Tỉnh ủy;
- Thường trực HĐND tỉnh;
- CT và các PCT UBND tỉnh;
- Báo Thừa Thiên Huế;
- Đài TRT;
- Công TTTT tỉnh;
- Công báo tỉnh;
- VP: CVP và các PCVP;
- Lưu: VT, CN.

**TM. ỦY BAN NHÂN DÂN**  
**KT. CHỦ TỊCH**  
**PHÓ CHỦ TỊCH**



*[Handwritten signature]*

**Nguyễn Thanh Bình**



Phụ lục

## HƯỚNG DẪN KỸ THUẬT

Tích hợp ứng dụng vào ứng dụng dịch vụ đô thị thông minh Hue-S

(Kèm theo Quyết định số 3071/QĐ-UBND ngày 26/11/2021

của Ủy ban nhân dân tỉnh Thừa Thiên Huế)

### I. TỔNG QUAN

#### 1. Phạm vi đối tượng áp dụng

Tài liệu này nhằm hướng dẫn các cơ quan, đơn vị phát triển các ứng dụng sẽ tích hợp lên nền tảng ứng dụng Hue-S.

#### 2. Các nền tảng hỗ trợ tích hợp

STT	Nền tảng ứng dụng bên thứ 3	Tích hợp được Hue-S	Ghi chú
1.	Android	Android	Nên dùng cho mục đích tích hợp các phần module native
2.	iOS	iOS	Nên dùng cho mục đích tích hợp các phần module native
3.	Flutter	Android iOS	Cần ưu tiên đóng gói ứng dụng tích hợp vào Hue-S bằng công nghệ Flutter

#### 3. Giải thích từ viết tắt

Trong hướng dẫn này các từ ngữ sau đây được hiểu như sau:

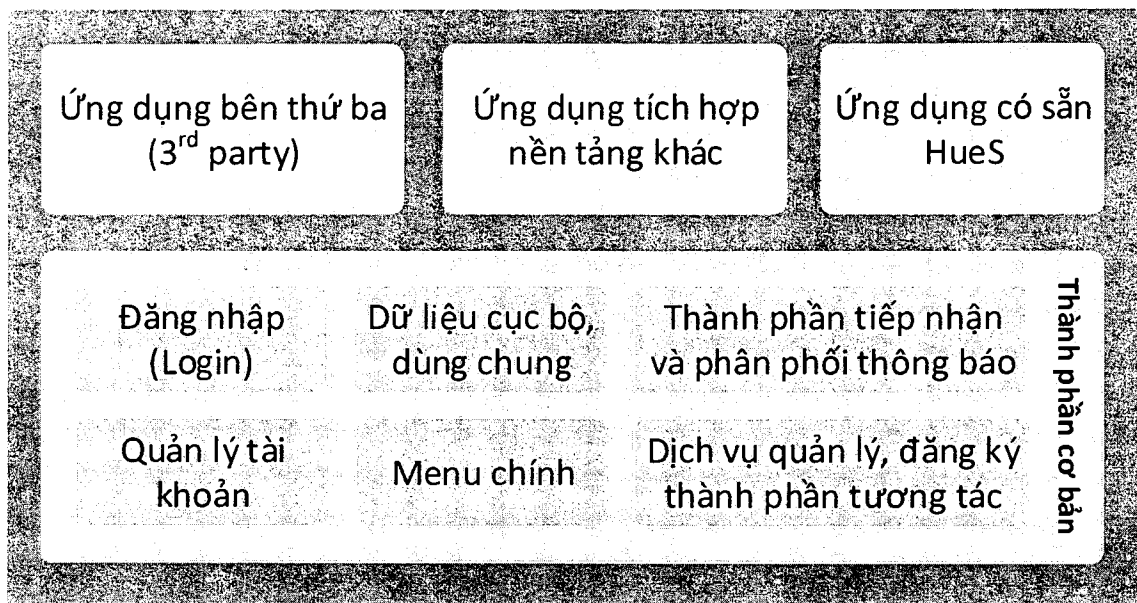
STT	Tên viết tắt	Mô tả
1.	Hue-S	Ứng dụng dịch vụ đô thị thông minh Thừa Thiên Huế trên các nền tảng di động
2.	NSD	Người sử dụng: người thao tác với ứng dụng Hue-S trên các thiết bị di động
3.	Android	Hệ điều hành Android của hãng Google dành cho các dòng thiết bị di động
4.	IOS	Hệ điều hành iOS của hãng Apple dành cho các dòng thiết bị di động của hãng Apple
5.	appId	Package name đối với Android, BundleId đối với iOS. Giá trị này cần được định danh duy nhất

## II. MÔ HÌNH KẾT NỐI

Các ứng dụng của bên thứ 3 hoàn toàn chủ động về mặt nghiệp vụ, tuy nhiên các thông tin tương tác cơ bản sẽ được trao đổi với ứng dụng chính Hue-S thông qua các lớp thành phần có thể tương tác được, với các phương thức như:

1. Component dùng chung
2. Sự kiện (Event) để trao đổi dữ liệu
3. Tạo các kênh kết nối

Ứng dụng Hue-S (Android/iOS)



Hình 1 Mô hình thành phần ứng dụng di động Hue-S

### III. QUY TRÌNH TÍCH HỢP CÁC ỨNG DỤNG VÀO NỀN TẢNG ỨNG DỤNG HUE-S

MỤC	TRÁCH NHIỆM	CÔNG VIỆC	TÀI LIỆU/ CÔNG CỤ
1.	Huế IOC	Tích hợp ứng dụng bên thứ 3	
2.	Đối tác phát triển ứng dụng	Tích hợp với ứng dụng khung mẫu	- Ứng dụng chạy thử với ứng dụng mẫu - Check list các thành phần tích hợp
3.	Huế IOC	Kiểm thử tích hợp ứng dụng mẫu	- Báo cáo kết quả kiểm thử
4.	Huế IOC Đối tác	Chuyển/Nhận mã nguồn ứng dụng tích hợp	- Biên bản giao nhận
5.	Huế IOC	Tích hợp và biên dịch ứng dụng Hue-S	- Ứng dụng đẩy lên kho test hoặc xuất bản thành file cài đặt
6.	Huế IOC	Kiểm thử tích hợp ứng dụng Hue-S	- Báo cáo kết quả test
7.	Huế IOC	Xuất bản lên kho ứng dụng	- Ứng dụng được xuất bản trên store: Google Play Store và Apple App Store
8.	Huế IOC	Kết thúc tích hợp	

## IV. ĐẶC TẢ CÁC THÀNH PHẦN KỸ THUẬT TÍCH HỢP

### 1. Đóng gói ứng dụng

- Ứng dụng Hue-S hỗ trợ tích hợp các ứng dụng được viết dưới dạng native cho từng hệ điều hành Android và IOS hoặc được viết bằng công nghệ đa nền tảng Flutter (Ưu tiên).

- Để tích hợp vào Hue-S thì ứng dụng cần được đóng gói thành 1 module có thể hoạt động độc lập riêng biệt cụ thể như sau:

+ Đối với Android thì sẽ đóng gói dưới dạng Android Library. Màn hình Activity chính của ứng dụng cần có tên là MainActivity và nằm ở thư mục mã nguồn gốc của ứng dụng. Ngoài ra ứng dụng cần cung cấp **ProGuard rules** cho quá trình **shrink, obfuscate** module.

+ Đối với IOS sẽ đóng gói dưới dạng Framework. ViewController chính của ứng dụng cần có tên là MainViewController.

+ Đối với đa nền tảng sử dụng Flutter thì cần đóng gói dưới dạng package. Root Widget của ứng dụng cần được đặt tên là MainApp

- Ứng dụng tích hợp cần phát triển dựa trên công nghệ đa nền tảng Flutter để đảm bảo hiệu năng tổng thể của ứng dụng Hue-S. Trong trường hợp ứng dụng tích hợp cần được phát triển dưới dạng Native thì sẽ cần trao đổi thêm để có hướng tích hợp phù hợp.

- Để module ứng dụng có thể giao tiếp được với các thành phần bên trong Hue-S (đăng nhập, lấy thông tin tài khoản, thông báo...) thì sẽ cần thêm SDK core của Hue-S. Ngoài ra, ứng dụng Hue-S cũng cung cấp khung căn bản của ứng dụng Hue-S để các đơn vị tiến hành tích hợp thử trước khi bàn giao hay cập nhật mã nguồn ứng dụng và khi bàn giao mã nguồn ứng dụng cần cung cấp dưới dạng đã tích hợp thành công vào khung Hue-S này.

- Mã nguồn ứng dụng phải được cung cấp ở dạng minh bạch có thể đọc được. Trong trường hợp cần giấu mã nguồn vì lý do bảo mật hoặc bảo vệ tài sản trí tuệ thì sẽ cần trao đổi trực tiếp để có phương hướng tích hợp phù hợp.

- Mã nguồn ứng dụng tích hợp sẽ được đẩy lên Git Repository do Hue-S cung cấp. Yêu cầu cần tạo các branch để giữ các phiên bản cũ (tối thiểu 5 phiên bản) để đề phòng trường hợp có lỗi nghiêm trọng không thể khắc phục sớm thì có thể tạm thời quay về bản cũ và cũng để phục vụ cho việc Hue-S cần kiểm tra giám sát sự thay đổi của các phiên bản.

- Trong quá trình Hue-S review code ứng dụng, nếu nhận ra có phần nào đó chưa phù hợp, chưa đảm bảo các yêu cầu chất lượng kỹ thuật thì có thể yêu cầu phía cung cấp ứng dụng tiến hành chỉnh sửa. Ngoài các yêu cầu về kỹ thuật, chất lượng ứng dụng được mô tả trong tài liệu thì về cơ bản các ứng dụng tích hợp vào Hue-S cần tuân thủ theo các yêu cầu, khuyến nghị về hiệu năng, bảo mật, trải nghiệm... do Apple và Google đưa ra dành cho việc phát triển ứng dụng di động.

### 2. Các thành phần tích hợp do ứng dụng Hue-S cung cấp

#### a. Tài khoản người dùng hệ thống

Thông tin tài khoản Hue-S bao gồm các trường dữ liệu sau (một số thông tin sẽ được bổ sung sau):

Tên trường	Kiểu dữ liệu	Bắt buộc	Mô tả
------------	--------------	----------	-------

Lớp (class) User			
username	String	x	Tên đăng nhập
name	String	x	Họ và tên người dùng
gender	Byte	x	Giới tính. 1: Nam; 2 Nữ;
birthday	String	x	Năm sinh người dùng định dạng dd/MM/yyyy
address	String		Địa chỉ thường trú của người dùng
email	String		Email
phone	String	x	Điện thoại di động
avatar			Url ảnh đại diện của người dùng

Ứng dụng tích hợp vào Hue-S có thể cần thêm một số trường thông tin khác liên quan đến dịch vụ của mình. Trong trường hợp này ứng dụng có thể tự bổ sung thêm màn hình cập nhật thông tin trong ứng dụng và tự quản lý các thông tin này (cập nhật, đồng bộ, lưu trữ...) mà không thông qua Hue-S. Tuy nhiên đơn vị cung cấp ứng dụng cần phải cung cấp API để Hue-S có thể truy xuất dữ liệu này khi cần thiết và cũng cần nêu rõ các trường dữ liệu cần thêm trước khi tích hợp ứng dụng. Lưu ý, cần tránh bắt buộc người dùng nhập các thông tin thêm của ứng dụng ngay từ lần đầu người dùng sử dụng ứng dụng nếu không thật sự cần thiết.

Ứng dụng Hue-S chỉ cho phép đăng nhập bằng tài khoản công dân Huế. Trong trường hợp phía cung cấp ứng dụng tích hợp đã có khối lượng dữ liệu tài khoản người dùng lớn và muốn tận dụng dữ liệu này thì có thể dựa vào thông tin người dùng tài khoản Hue-S (ví dụ số điện thoại, địa chỉ email) để kiểm tra xem người dùng này đã tồn tại trên hệ thống hay chưa và sẽ gợi ý người dùng tiến hành đồng bộ dữ liệu giữa 2 tài khoản công dân Huế và tài khoản của phía cung cấp ứng dụng tích hợp.

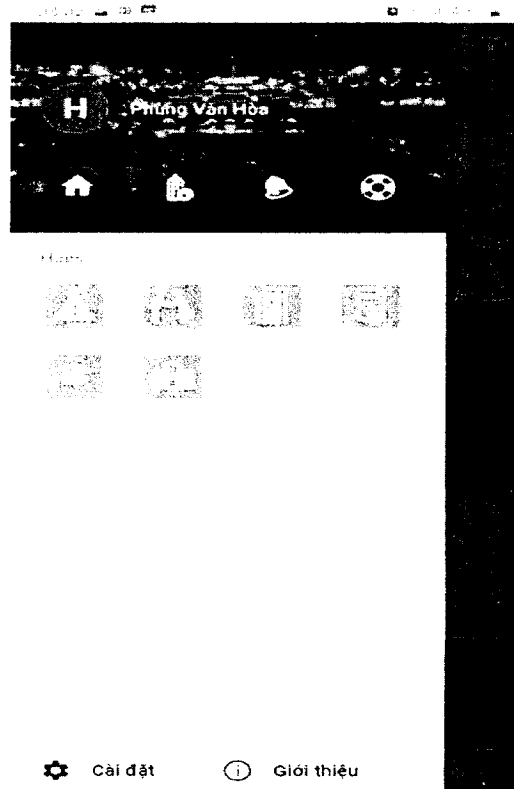
Thư viện tích hợp Hue-S sẽ bao gồm aAPI tài khoản để các ứng dụng tích hợp có thể lấy được thông tin tài khoản hiện tại đang đăng nhập và token xác thực của tài khoản (xác thực thông tin người dùng trên hệ thống quản lý tài khoản công dân Huế) dành cho ứng dụng tích hợp. Lưu ý, các ứng dụng tích hợp khác nhau sẽ có token xác thực người dùng khác nhau.

#### **b. Hệ thống Menu điều hướng chính (Navigation Menu)**

Là hệ thống Menu điều hướng thống nhất của ứng dụng Hue-S. Từ các ứng dụng tích hợp đều có thể gọi Menu này nhằm thực hiện các điều hướng (navigative): về màn hình chính của ứng dụng, các ứng dụng ngang cấp khác, các chức năng thông tin cơ bản. Việc gọi menu chính sẽ được cài đặt ở nút menu (dạng Hamburger ≡ ) hoặc vuốt từ cạnh trái màn hình sang phải. Các ứng dụng tích hợp chủ động thiết lập menu hamburger tại các vị trí cần thiết của ứng dụng.

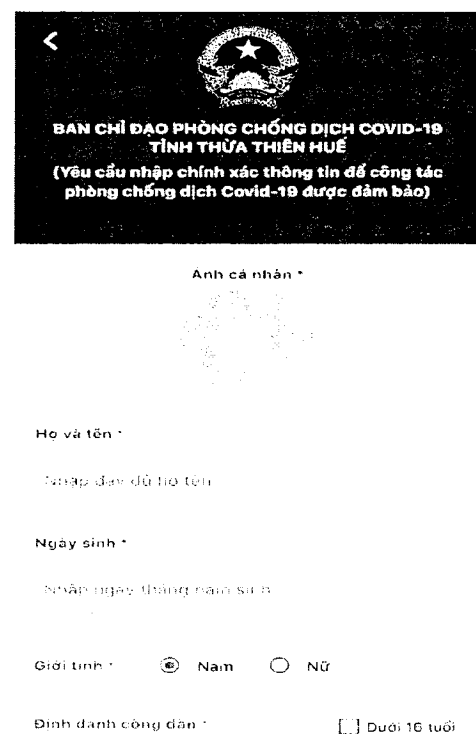
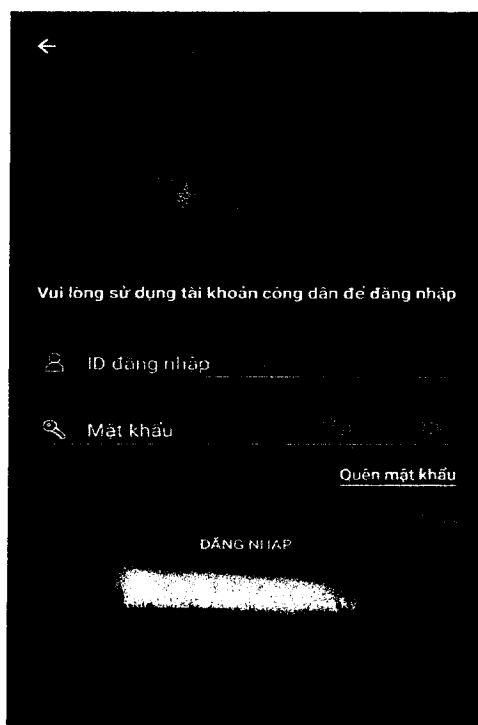
Navigation Menu này được cung cấp trong SDK tích hợp Hue-S. Ngoài các Menu điều hướng cố định của Hue-S thì các đơn vị tích hợp cũng có thể bổ sung thêm các Menu điều hướng riêng của ứng dụng thông qua API.

Ngoài ra trong Navigation Menu còn có 2 Menu cố định sẵn dành riêng cho từng ứng dụng tích hợp là **Hướng dẫn** (dành cho việc điều hướng đến màn hình hướng dẫn cũng như giới thiệu ứng dụng nếu có) và **Cài đặt** (dành cho việc điều hướng đến màn hình cài đặt riêng của ứng dụng tích hợp nếu ứng dụng tích hợp có các cấu hình cho phép người dùng tùy chỉnh). Các ứng dụng tích hợp chủ động ẩn/hiện 2 Menu này tùy theo ứng dụng có các mục trên hay không thông qua API.



### c. Màn hình đăng nhập, đăng ký

Ứng dụng tích hợp khi cần thực hiện các chức năng yêu cầu phải chứng thực người dùng có thể tiến hành điều hướng đến màn hình đăng nhập hoặc màn hình đăng ký thông qua API do Hue-S cung cấp. Sau khi người dùng đăng nhập thành công thì tất cả các ứng dụng tích hợp sẽ nhận được sự kiện đăng nhập bao





gồm thông tin về tài khoản của người dùng vừa đăng nhập

#### d. Thành phần tiếp nhận và phân phối thông báo (Notification)

Ứng dụng Hue-S sẽ là nơi tiếp nhận tất cả các thông báo được đẩy đến, và sau đó Hue-S sẽ phân phối thông báo cho ứng dụng tích hợp tương ứng

#### e. Chia sẻ dữ liệu (bao gồm local data và remote)

Hue-S sẽ cũng cấp kênh trao đổi dữ liệu để các ứng dụng tích hợp có thể thực hiện chia sẻ dữ liệu cho nhau cũng như cung cấp dữ liệu cho Hue-S khi cần thiết.

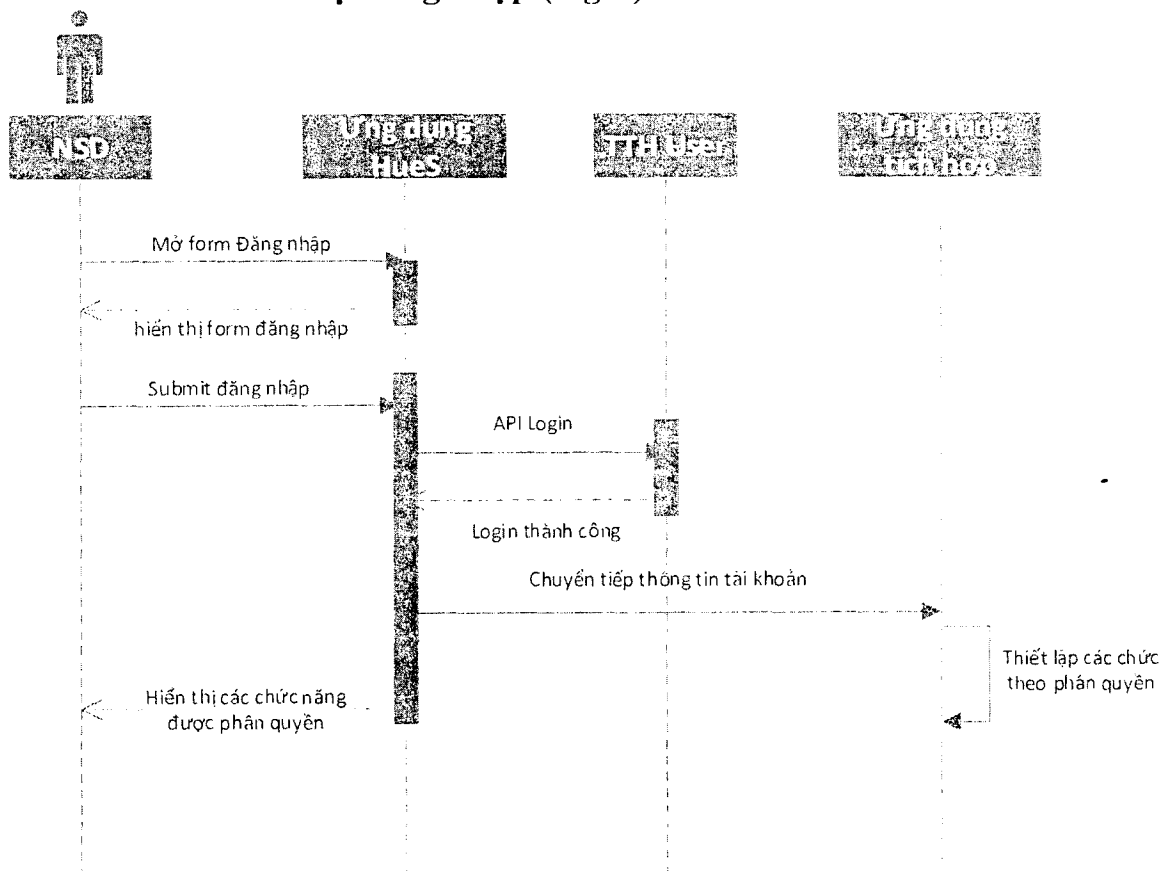
### 3. Tích hợp tài khoản

#### a. Nguyên tắc

Mỗi khi người dùng đăng nhập hoặc đăng xuất Hue-S sẽ phát ra 1 sự kiện thông báo cho toàn bộ các ứng dụng tích hợp được biết. Các ứng dụng cần dựa vào sự kiện này để có cách hành động phù hợp nếu cần. Ví dụ như xóa các dữ liệu cache cá nhân của tài khoản vừa đăng xuất, hủy đăng 1 số topic firebase liên quan đến tài khoản vừa đăng xuất, dừng các request liên quan đến tài khoản vừa đăng xuất...

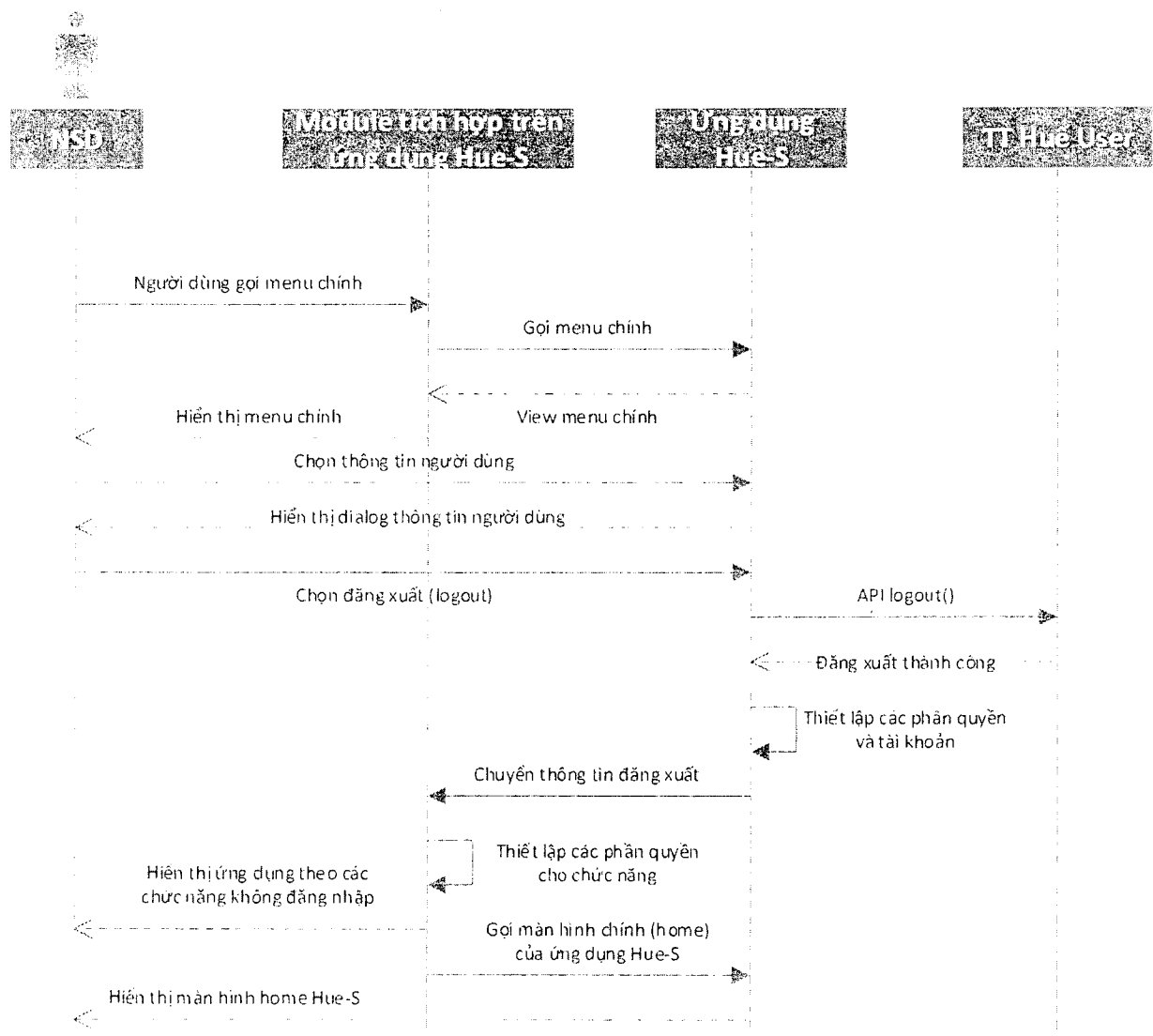
Các tác vụ của các ứng dụng tích hợp cần thực hiện khi nhận sự kiện đăng nhập/dăng xuất nên cố gắng thực hiện nhanh. Các tác vụ lâu hoặc nặng nên chờ đến khi người dùng vào app cụ thể mới tiến hành thực hiện.

#### b. Sơ đồ tuần tự đăng nhập (login)



Hình 2 Sơ đồ tuần tự đăng nhập từ hệ thống Hue-S

#### c. Sơ đồ tuần tự đăng xuất (logout)



Hình 3 Sơ đồ tuần tự quá trình đăng xuất ứng dụng trên các module tích hợp

#### 4. Quy tắc đặt tên tránh xung đột

Các ứng dụng tích hợp vào Hue-S phải cần đảm bảo có các giá trị tên riêng biệt sau đây:

- Tên miền của đơn vị cung cấp ứng dụng tích hợp
- Tên ứng dụng
- Prefix name (dùng cho đặt tên file, tag...) riêng biệt. Ví dụ ứng dụng có prefix name là cmv và cần đặt giá trị tag là **waitUpload** thì giá trị cuối cùng của tag sẽ là **cmvWaitUpload** hoặc **cmv\_waitUpload**...

Về cơ bản để tránh xung đột thì mỗi khi ứng dụng cần 1 giá trị để sử dụng cho đặt tên File hoặc Tag cho các api thì giá trị đó cần bao gồm package name (hoặc Bundle Identifier) hoặc prefix name (trong trường hợp tên hoặc tag cần ngắn gọn) riêng biệt mà ứng dụng đã đăng ký trước khi tích hợp vào Hue-S. Trong đó có các trường hợp được quy định cụ thể như sau.

##### a. Android

- Module package name

Các module được tạo (bao gồm cả module chính tích hợp và các module tự tạo khác) cần được đặt tên package theo quy tắc **topleveldomain.applicationname** để tránh xung đột với các module khác

Ví dụ công ty có tên miền là **example.com** và chức năng feedback là upload sẽ được đặt tên **com.example.feedback**

- File, Folder (bao gồm cả database, SharedPreferences) được sinh ra trong quá trình sử dụng ứng dụng cần là con của thư mục gốc cha có đường dẫn như sau:

- + Internal Storage (bao gồm cả đường dẫn cache và data):  
đường\_dẫn\_gốc\_internal/{Tên\_ứng\_dụng}

- + External Storage: đường\_dẫn\_gốc\_external/HueS/{Tên\_ứng\_dụng}  
(Lưu ý: Cần kiểm tra xem thư mục Hue-S đã tồn tại chưa)

- Resource (drawable, string, style....)

Cần đặt tên có prefix riêng biệt và quy định giá trị prefix này trong gradle thông qua key “resourcePrefix” (có thể lấy tên viết tắt của app làm prefix)

- Các name, tag cần phải riêng biệt như Intent Action, Content Uri Authority, WakeLock Tag, Permission Name...

Đặt tên cần bao gồm cả package name của module để tránh xung đột

- Universal Links

Phần domain của link cho tất cả các ứng dụng có giá trị là domain của Hue-S. Các ứng dụng tích hợp sẽ sử dụng phần path của link để phân biệt link giữa các ứng dụng. Giá trị phần path này cần đặt riêng biệt để không xung đột với các ứng dụng khác

- Notification channel Id

Id kênh thông báo cần được đặt theo quy tắc {package\_name}\_{id}. Ví dụ **com.example.feedback\_hotSale**

- Process name:

Đặt tên process cần bao gồm prefix name riêng biệt của ứng dụng. Lưu ý hiện tại Hue-S chưa hỗ trợ app tích hợp tạo thêm process.

## **b. IOS**

- File, Folder được sinh ra trong quá trình sử dụng ứng dụng cần là con của thư mục gốc cha có đường dẫn như sau:  
đường\_dẫn\_gốc\_internal/{Tên\_ứng\_dụng}. Trong đó có 1 số lưu ý như sau:

- + Đường\_dẫn\_gốc\_internal là đường dẫn đến các loại thư mục lưu trữ nội bộ gốc của ứng dụng bao gồm: **Documents, Library, Library/Caches, tmp**.

- + Các File, Folder được lưu trữ trong **Documents** sẽ được chia sẻ để người dùng có thể truy xuất khi cần thiết (qua ứng dụng quản lý Files). Vì vậy không nên lưu trữ các nội dung mà không muốn người dùng truy cập trực tiếp ở thư mục **Documents**

- UserDefaults

Bắt buộc khởi tạo UserDefaults kèm suiteName bao gồm prefix name riêng biệt của ứng dụng.

- Local Notification(UNNotification)

Trong trường hợp không sử dụng UUID cho identifier mà cần tùy biến tên thì cần đặt tên bao gồm bundle identifier

- NotificationCenter

Giá trị tên được dùng để post cần bao gồm prefix name riêng biệt của ứng dụng

- KeyChain

Giá trị key phải bao gồm prefix name riêng biệt của ứng dụng.

- Framework bundle id

Cách đặt giá trị bundle id cho framework (bao gồm cả framework chính tích hợp và các framework tự tạo khác) giống với đặt tên package module cho Android

### c. Flutter

- Package name: Để tránh xung đột tên cần đặt tên cho package (library) ngắn gọn có liên quan đến tên của ứng dụng và bao gồm prefix name riêng biệt của ứng dụng đã đăng ký từ đầu. Các plugin, package flutter tự tạo khác thì cần đặt tên bao gồm Prefix Name riêng biệt của ứng dụng.

- Method Channel: Các tên method channel được sử dụng cần bao gồm package name có tên miền ngược của ứng dụng.

- Route: Các ứng dụng tích hợp quy định tên route cần có prefix là base route có dạng `/ {package_name}` (truy cập route này sẽ vào màn hình chính ứng dụng tích hợp). Ví dụ App A tích hợp vào Hue-S có các route như sau `/appA/featureA`, `/appA/featureB`

## 5. Deep Links

### a. Android

Chỉ cần khai báo Activity tương ứng với Link của ứng dụng trong file **AndroidManifest**

### b. IOS

Chỉ sử dụng Universal Links không sử dụng Url Scheme.

Khi ứng dụng Hue-S kích hoạt từ Universal Links, Hue-S sẽ gọi Class xử lý link của ứng dụng liên quan (Class này sẽ phải đặt tên theo quy ước từ trước) dựa vào phần path của link.

## 6. Tích hợp notification

### a. Local Notification

Đối với IOS thì chỉ cần đặt giá trị identifier () theo quy ước đã được nêu ở phần đặt tên sao cho tránh xung đột

Đối với Android sẽ có 2 trường hợp cần tạo Local Notification là khi nhận được thông báo data Firebase do Hue-S làm trung gian chuyển do ứng dụng và ứng dụng tự tạo thông báo local (Ví dụ người dùng đặt lịch nhắc nhở). Việc tạo thông báo local trên Android cần tuân thủ 1 số quy tắc sau đây:

- Cần tạo các kênh thông báo riêng biệt cho từng loại thông báo khác nhau (Notification Channel) để người dùng có thể tùy chọn bật tắt thông báo không cần thiết với bản thân.

- Mỗi ứng dụng cần tích hợp vào Hue-S sẽ được cung cấp sẵn nhóm kênh thông báo (Notification channel group) trước khi tiến hành tích hợp ứng dụng. Các ứng dụng tích hợp bắt buộc phải sử dụng groupId của nhóm thông báo cung cấp sẵn để tạo kênh thông báo.

- Thông báo local bắt buộc phải được khởi tạo kèm theo tag là package name của ứng dụng còn id có giá trị tùy ý: **notify (String package\_name, int id, Notification notification)**

### b. Remote Notification

Khi các ứng dụng tích hợp cần đẩy thông báo cho người dùng, thay vì gửi trực tiếp lên firebase sẽ được đưa tập trung về Server của Hue-S và Server Hue-

S sẽ thực hiện việc push notification lên Firebase. Việc này nhằm kiểm soát tần suất đẩy thông báo, giảm tính phức tạp trong cấu hình thông báo ứng dụng và giúp cho ứng dụng Hue-S có thể lấy danh sách thông báo (xem lại các thông báo đã đẩy) nhanh chóng từ 1 nguồn duy nhất.

**POST** <https://tuongtac.thuathienhue.gov.vn/dichvu/push>

- Header:

apikey: {key} (giá trị này sẽ được cấp cho ứng dụng trước khi tích hợp)

Content-Type: application/json

- Body:

```
{
  "to": "/topics/caros_hotSale",
  "title": "Tiêu đề thông báo",
  "message": "Nội dung body thông báo",
  "data": {
    "id": "1",
    "name": "Tom",
    "photo": "https://pbs.twimg.com/profile_images/735509975649378305/B81JwLT7.jpg"
  },
  "priority": "normal",
  "android": {
    "channel_id": null
  }
}
```

+ to: Xác định thông báo này sẽ gửi cho ai. Giá trị có thể là tên topic (/topics/{tên\_topic}) hoặc username của người dùng cụ thể muốn gửi. Hiện tại chưa hỗ trợ gửi theo nhóm. Lưu ý tên topic cần thêm tiền tố prefix name riêng biệt của ứng dụng để tránh xung đột. Ví dụ cần gửi thông báo đến topic có tên hotSale (to: /topics/hotSale) và ứng dụng có prefix name là caro thì tên topic sẽ là caro\_hotSale

+ priority: Chỉ định mức độ quan trọng của thông báo gồm 2 giá trị normal và high. Lưu ý đối với cái thông báo không quan trọng, mang tính quảng cáo, giới thiệu, phạm vi đối tượng nhắm đến là cộng đồng thì không đặt giá trị high.

+ title: Tiêu đề thông báo. Tiêu đề nên mang tính đặc trưng của ứng dụng nếu không thì sử dụng tên ứng dụng làm tiêu đề thông báo.

+ message: Nội dung thông báo. Nội dung này nên ngắn gọn, cụ thể.

+ data (có thể null): Dữ liệu gửi kèm thông báo có kích thước không vượt quá 3 KB. Trong trường hợp vượt quá 3 KB thì có thể dùng gzip để nén dung lượng text data

+ channel\_id (có thể null): Chỉ định notification channel id của Android.

Mặc định thì thông báo Android từ firebase sẽ được đẩy vào kênh thông báo có id là appId của ứng dụng tích hợp. Nếu channel\_id có giá trị thì thông báo sẽ được đẩy vào kênh thông báo có id là {appId}\_{id}. Các ứng dụng tích hợp cần chủ động tạo các kênh thông báo của mình.

### c. Firebase

Hiện tại chỉ hỗ trợ tích hợp Crashlytics cho việc log lỗi. Trong 1 khoảng thời gian định kỳ nhất định Hue-S tiến hành rà soát lỗi ứng dụng và sẽ gửi các

lỗi liên quan đến ứng dụng cho bên cung cấp ứng dụng để tiến hành vá lỗi. Các dịch vụ khác về sau sẽ tiến hành tích hợp thông qua Firebase BigQuery và Firebase Api nếu được hỗ trợ

## 7. Lưu trữ và đồng bộ dữ liệu

Mỗi ứng dụng sẽ sử dụng file database riêng để tránh xung đột dữ liệu. Vì sử dụng database riêng nên mỗi ứng dụng có thể tùy ý sử dụng loại database (file, sql, nosql) Hue-S sẽ phát 1 sự kiện khi người dùng mở ứng dụng Hue-S để các ứng dụng tích hợp có thể tiến hành đồng bộ các dữ liệu quan trọng, cần thiết đối với người dùng. Các ứng dụng tích hợp nhận sự kiện này thông qua việc kế thừa class **AppDelegateImpl**. Tuy nhiên cần tránh thực hiện nhiều request hoặc các tác vụ lâu nếu không thật sự cần thiết trong thời gian này.

Nếu ứng dụng có các tác vụ định kỳ cần thực hiện ở background (khi ứng dụng không ở trạng thái active đối với IOS hoặc thậm chí khi ứng dụng không được bật trước đó đối với Android) thì cần trao đổi thêm trực tiếp để có phương án tích hợp phù hợp.

Android và IOS có cung cấp nhiều loại vùng nhớ cho từng loại dữ liệu khác nhau. Cần phải lựa chọn đúng loại vùng nhớ theo khuyến nghị của Apple và Google vì Hue-S sẽ chủ động dọn dẹp các vùng nhớ (tùy theo từng hoàn cảnh sẽ chọn các vùng nhớ khác nhau) của các ứng dụng khi cần thiết. Ngoài ra dung lượng bộ nhớ trong cho các vùng nhớ lâu dài (không phải cache hoặc temporary) tối đa mà các ứng dụng được sử dụng là 100MB.

## 8. Bảo mật

- Hạn chế lưu trữ các dữ liệu mang tính nhạy cảm của người dùng nếu không cần thiết. Nếu có lưu trữ thì dữ liệu cần được mã hóa bởi cơ chế sinh khóa bảo mật cao hoặc lưu trữ trong các vùng nhớ bảo mật cao do hệ điều hành cung cấp (Keychain đối với ios và Keystore đối với Android). Đối với database (sql, nosql) thì có thể sử dụng các thư viện hỗ trợ mã hóa là sqlcipher hay Realm.

- Các request network nếu có gửi nhận dữ liệu hay thao tác mang tính nhạy cảm thì bắt buộc phải sử dụng HTTPS. Server luôn phải xác thực, kiểm tra tính hợp lệ của request, tuyệt đối không tin tưởng vào client.

- Tránh trả dữ liệu dư thừa cho người dùng để không lãng phí dung lượng của người dùng và tăng tính bảo mật dữ liệu. Những dữ liệu người dùng không nên thấy thì không trả về cho dù có chặn hiển thị trên app.

- Đối với các chức năng, thao tác cần tính bảo mật cao trên Android thì khuyến nghị nên kiểm tra độ tin cậy của thiết bị thông qua SafetyNet Api. Phần này sẽ do Hue-S cung cấp trung gian thông qua API tích hợp.

- Webview chỉ được sử dụng để hiển thị nội dung html tĩnh. Không được sử dụng Webview để hiển thị hoặc tải nội dung của các bên khác. Trong trường hợp cần tải nội dung trang web ngoài thì sử dụng Chrome Custom Tabs hoặc mở trình duyệt của thiết bị

- Đối với các file được upload từ ứng dụng cần đổi tên file trước hoặc sau khi upload để tránh trường hợp lộ thông tin người dùng từ tên file trừ khi có yêu cầu giữ tên từ người dùng hoặc việc giữ tên là cần thiết đối với người dùng. Đối với các file ảnh và video khi tải lên server cần phải xóa các thông tin metadata có liên quan đến người dùng (ngày tạo, chỉnh sửa, vị trí, thông số camera...)

- Khi người dùng đăng xuất tài khoản thì ứng dụng cần xóa hết các dữ liệu liên quan đến tài khoản. Nếu cần giữ lại thì phải mã hóa để chỉ khi tài khoản đó đăng nhập lại mới có thể khai thác dữ liệu được.

## **9. Hiệu năng**

- Các request network thông thường (tổng dữ liệu gửi/nhận nhỏ hơn 100KB và không thực hiện các tính toán phức tạp) có thời gian trung bình không quá 2 giây

- Tuyệt đối không thực hiện các tác vụ nặng hoặc mất nhiều thời gian (network, truy xuất file/database...) ở **thread UI**

- Cố gắng kiểm soát memory ứng dụng để không bị đẩy lên quá nhanh hoặc quá cao gần ngưỡng crash. Hạn chế tối đa memory leak. Ngoài ra cần hạn chế việc tạo thêm process trên Android và mỗi khi kết thúc process thì nên chủ động kết thúc (kill) process nếu process không được cần mở thường xuyên hoặc không phải process quan trọng.

- Khi ứng dụng rơi vào trạng thái không kích hoạt thì nên dừng các tác vụ không cần thiết (nhất là các request network hoặc tác vụ nặng, sensor) để giảm hao phí dung lượng pin, cpu. Ngoài ra, khi người dùng thoát khỏi màn hình chức năng cũng cần phải dừng các tác vụ không cần thiết được phát sinh ở màn hình này.

- Tuyệt đối không lạm dụng đẩy thông báo ở dạng im lặng (data Android và silent IOS) để kích hoạt thiết bị active cho mục đích đồng bộ dữ liệu. Nếu cần dựa vào thông báo để đồng bộ dữ liệu lớn và thường xuyên thì cần trao đổi thêm.

- Tuyệt đối không tự tạo các lịch task chạy ngầm bên Android (bao gồm cả Service) hoặc các background task bên IOS để đồng bộ dữ liệu. Như đã ghi trên phần lưu trữ và đồng bộ dữ liệu, việc này cần trao đổi thêm.

- Cố gắng cache dữ liệu để tránh việc request network liên tục trong thời gian ngắn. Ngoài ra nên tạo các phiên bản nhỏ của dữ liệu. Ví dụ như Ảnh thì sẽ có ảnh thumb để hiển thị ở danh sách thay vì sử dụng ảnh đầy đủ. Danh sách bài viết thì chỉ cần 1 vài thông tin như tiêu đề, tóm tắt, ngày đăng chứ không cần toàn bộ nội dung bài viết. Như vậy không chỉ giảm dung lượng dữ liệu mà còn giảm thiểu CPU sử dụng để xử lý dữ liệu

- Đối với các tác vụ xử lý nặng thì nên có các phương hướng xử lý, các hạn chế ràng buộc dữ liệu cho từng phân khúc cấu hình thiết bị nếu có thể. Ví dụ với tác vụ nén ảnh thì các thiết bị có memory lớn sẽ cho phép thực hiện 3 tác vụ nén ảnh chạy song song, còn với các thiết bị có dung lượng memory nhỏ thì chỉ cho phép thực hiện 1 tác vụ nén trong 1 thời điểm.

## **10. Tích hợp các dịch vụ ứng dụng di động**

### **a. Thanh toán trong ứng dụng (in-app purchase)**

Thanh toán trong ứng dụng Hue-S sẽ không thông qua các dịch vụ thanh toán của bên thứ 3 nào khác (kể cả dịch vụ thanh toán trong ứng dụng của Apple và Google) mà sử dụng cổng thanh toán riêng. Các ứng dụng tích hợp sẽ sử dụng các api thanh toán do Hue-S cung cấp để tiến hành thanh toán trong ứng dụng.

Trong trường hợp ứng dụng có các nội dung số cần thanh toán mà theo chính sách phát triển ứng dụng bắt buộc phải sử dụng dịch vụ **in-app purchase** của Google/Apple thì phải tự tìm cách chuyển đổi phù hợp.

**b. Bản đồ**

Hue-S sẽ cung cấp cho các đơn vị phát triển ứng dụng dịch vụ bản đồ miễn phí bao gồm sdk và dữ liệu bản đồ trong phạm vi tỉnh Thừa Thiên Huế. Các ứng dụng tích hợp vào Hue-S bắt buộc sẽ phải sử dụng bản đồ này. Trong trường hợp dịch vụ bản đồ của Hue-S chưa đáp ứng đủ yêu cầu thì sẽ tiến hành trao đổi thêm để có phương án tích hợp bản đồ phù hợp.

## **V. YÊU GIAO DIỆN ỨNG DỤNG TÍCH HỢP**

### **1. Bố cục màu sắc**

Cần dựa theo các guideline của Apple (ios) và Google material (android). Trong đó có một số yêu cầu cụ thể như sau:

- Cần nhất quán giao diện ứng dụng trên các nền tảng khác nhau (Android và IOS) để người dùng có thể dễ dàng sử dụng ứng dụng trên nhiều thiết bị di động. Đối với các giao diện dành cho các thiết bị di động quá khác biệt (đồng hồ thông minh, tivi thông minh...) hay trên máy tính thì cũng cần có những sự tương đồng nhất định với giao diện trên điện thoại.

- Không sử dụng quá 3 màu chính (không tính màu đen, trắng và gradient) và cần nhất quán trong việc sử dụng màu. Tuyệt đối không lạm dụng màu sắc khiến cho người dùng khó đọc được nội dung trừ trường hợp màn hình có ít nội dung chữ cần đọc

- Không sử dụng nhiều loại font chữ trong ứng dụng. Cần sử dụng các loại font, kích thước font và màu chữ sao cho dễ đọc.

- Sử dụng các icon đơn giản và hiện đại ví dụ như material icon của google

- Bố cục sắp xếp các nội dung ứng dụng cần gọn gàng, dễ nhìn trên nhiều kích thước màn hình và scaling. Phải có sự phân tách rõ ràng giữa các nhóm nội dung trên màn hình

- Với mỗi màn hình chức năng, các thành phần tương tác được định hướng là chức năng chính tại màn hình đó thì cần được thể hiện nổi bật hơn (bởi màu sắc hoặc kích thước) so với các thành phần khác.

- Ứng dụng cần có navigation menu bên trái. Ngoài các thông tin bắt buộc của Hue-S bao gồm: header, danh sách ứng dụng, giới thiệu ứng dụng hiện tại (thông tin ứng dụng, hướng dẫn sử dụng...) và cài đặt (nếu có) thì vẫn có thể tùy ý thêm các menu điều hướng riêng của ứng dụng vào navigation.

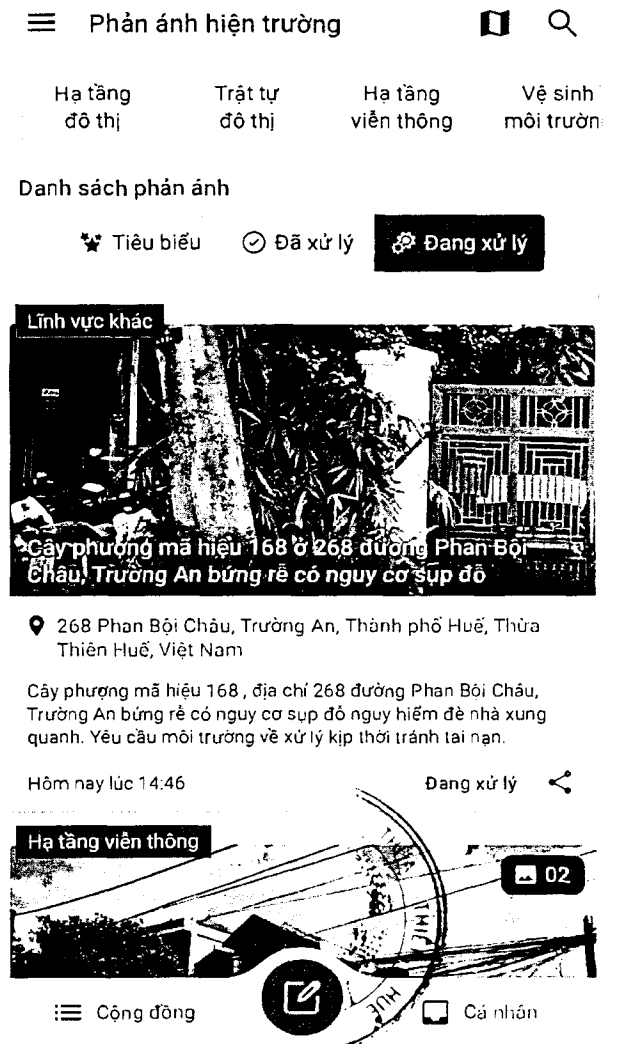
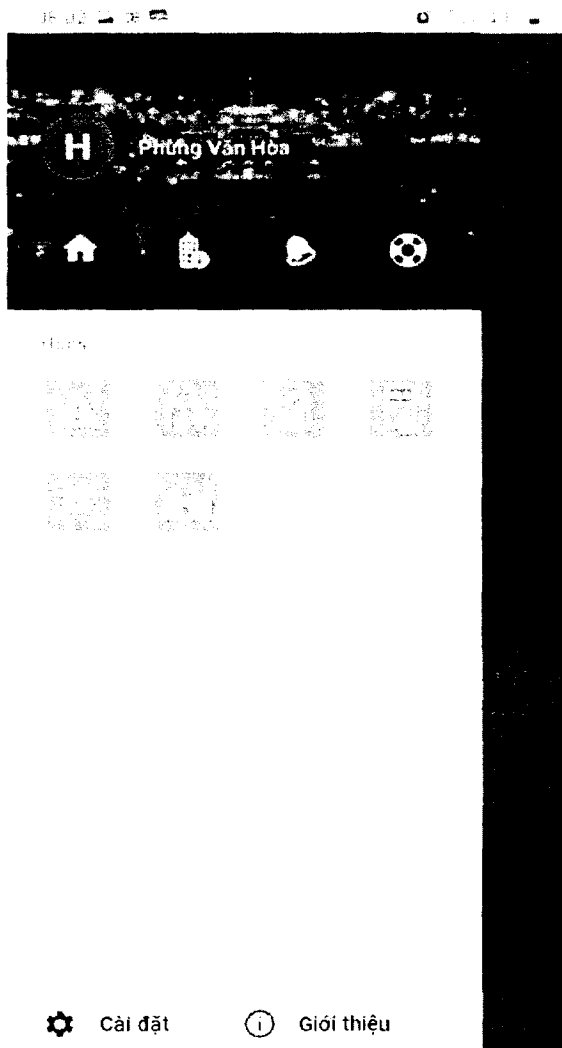
- Hạn chế nhét các điều hướng, chức năng quan trọng hoặc được dùng thường xuyên của ứng dụng ở navigation menu. Cố gắng thể hiện các chức năng đó ở màn hình chính

- Không nên thể hiện quá nhiều nội dung trong 1 màn hình nếu không thật sự cần thiết. Nên chia nhỏ các nhóm nội dung ra nhiều màn hình để người dùng dễ đọc và thao tác. Trong trường hợp thiết kế cho máy tính bảng thì 1 màn hình giao diện của máy tính bảng có thể là tổng hợp nhiều màn hình giao diện điện thoại.

- Màn hình đầu ứng dụng cần có tên ứng dụng (tên có thể biểu diễn cả dạng text hoặc hình ảnh miễn sao có thể nhận dạng được đây là ứng dụng gì) và



nút hamburger để mở navigation menu (cũng có thể mở bằng cách vuốt từ mép trái màn hình).



## 2. Trải nghiệm

- Không sử dụng webview để thể hiện màn hình chức năng trừ trường hợp cần hiển thị nội dung html tĩnh. Nên lập trình các màn hình chức năng dưới dạng native hoặc đa nền tảng flutter để đem lại trải nghiệm tốt cho người dùng và đảm bảo tính bảo mật.

- Cố gắng tối ưu để tránh việc giật, khựng khung hình trong quá trình sử dụng ứng dụng.

- Các thao tác load dữ liệu hoặc xử lý dữ liệu nên phải có nút dừng để người dùng có thể dừng thao tác (khi cảm thấy lâu hoặc không cần nữa). Ngoài ra nên cố gắng hạn chế các loading indicator toàn màn hình khi không thật sự cần thiết.

- Các nút, chữ...cho phép người dùng tương tác (nhấn, kéo...) phải có kích thước vùng bấm (tương tác) đủ lớn để người dùng dễ thao tác

- Thông báo lỗi thân thiện và dễ hiểu đồng thời kèm hướng giải quyết nếu có thể.

- Các nội dung text chi tiết nội dung nên cho người dùng có thể copy được

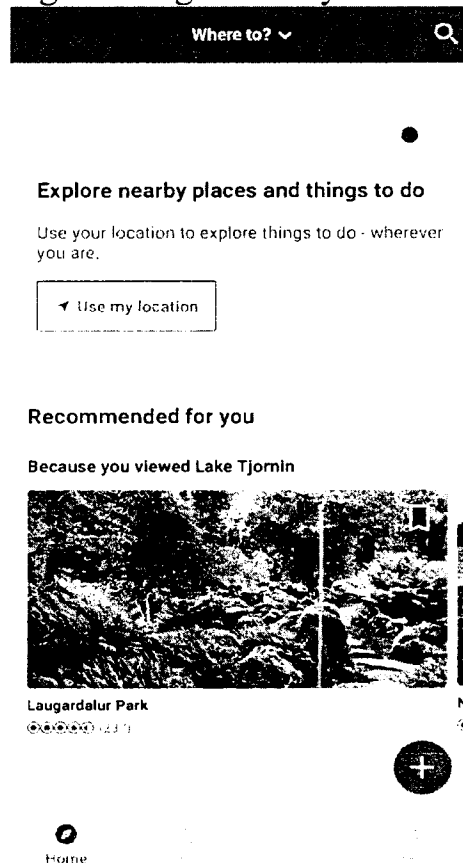
- Màn hình xem chi tiết nội dung phải cho người dùng có thể mở bằng trình duyệt để xem ở dạng web nếu có trang web

- Yêu cầu quyền:

+ Thực hiện yêu cầu quyền theo hoàn cảnh và phải giải thích vì sao cần quyền

+ Khi cần người dùng nhảy tới Settings (yêu cầu quyền bị deny, bật GPS...) cần giải thích rõ và cho phép người dùng quyết định có đến màn hình Settings hay không

+ Khi vào 1 màn hình nội dung và màn hình đó cần yêu cầu 1 quyền nào đó. Nếu quyền đó không quá quan trọng (hoặc đã từng bị deny nhiều lần) thì không nên hiển thị dialog yêu cầu quyền ngay khi vào màn hình để tránh việc làm phiền người dùng. Có thể hiển thị message yêu cầu cấp quyền cùng với nội dung của màn hình đó để người dùng cảm thấy cần thì mới bấm cấp quyền.



- Khi vào màn hình cần bật GPS thì không nên bắt buộc người dùng phải bật nếu không quá cần thiết. Cần hiển thị message lý do vì sao cần bật GPS. Đối với Android thì nên bật thông qua api google chứ không nhảy đến Settings.

- Cache các dữ liệu cần thiết với người dùng hoặc các dữ liệu ít thay đổi để có thể sử dụng offline hoặc cho hiện trước dữ liệu cache trước khi request dữ liệu từ mạng để tạo cảm giác tải dữ liệu nhanh (hình ảnh, ghi chú đánh dấu... thông tin tài khoản...).