

LỜI MỞ ĐẦU

Trong thời đại công nghệ thông tin đang bùng nổ mạnh mẽ thì các quốc gia trên thế giới trong đó có nước Việt Nam chúng ta đã và đang áp dụng công nghệ thông tin vào mọi mặt của đời sống – xã hội.

Áp dụng công nghệ thông tin vào các ngành, các lĩnh vực cuộc sống trở nên rất cần thiết, đặc biệt là là các ứng dụng tin học trong lĩnh vực quản lý. Nó giúp cho con người nắm bắt, sử dụng thông tin chính xác và nhanh chóng, từ đó đạt được chất lượng hiệu quả cao.

Việc Samsung đang phát triển mạnh mẽ ở Việt Nam, các mẫu điện thoại được Samsung sản xuất ngày càng nhiều. Do đó việc trưng bày các mẫu điện thoại mới cho người dùng trải nghiệm là rất nhiều, việc quản lý và bảo vệ việc tránh đánh cắp điện thoại là một vấn đề đang được đặt ra.

Hệ thống an ninh được cài đặt để ngăn chặn trường hợp trộm cắp và sặc ố định các thiết bị được trưng bày trong cửa hàng. Hệ thống an ninh có thể rung chuông báo động cho nhân viên cửa hàng để ngăn chặn trường hợp trộm cắp và có yếu tố hình thức nhỏ hơn để ít ảnh hưởng đến trải nghiệm của khách hàng. Tuy nhiên, trong các cửa hàng vận hành và bán lẻ, hệ thống bảo mật hình thức công kênh và tốn kém chi phí được triển khai. Do đó, nó không chỉ cản trở trải nghiệm của khách hàng mà còn thể hiện tính không nhất quán trong các cửa hàng và gây tốn kém chi phí. Hơn nữa, hệ thống an ninh hiện tại có sự phụ thuộc vào các nhà cung cấp bảo mật bằng hộp điều khiển.

Như tìm hiểu của em hiện nay, hai ứng dụng bảo vệ điện thoại và ứng dụng phát video ở Samsung là hai ứng dụng hoàn toàn riêng biệt. Nhằm khắc phục sự bất tiện khi cần cài hai ứng dụng gây tốn thời gian và không khách quan trong việc hiển thị trên khay ứng dụng do đó em đã thực hiện ghép hai ứng dụng trên vào làm một ứng dụng. Không chỉ vậy, nhằm khắc phục những vấn đề chưa tối ưu ở trên nên em

chọn đề tài: “*Phát triển ứng dụng theo dõi và bảo vệ các mẫu điện thoại thông minh của Samsung tại các cửa hàng bán lẻ*” nhằm giúp người sử dụng có một sự trải nghiệm hoàn hảo vào tiện lợi nhất, luôn đảm bảo hiệu năng hệ thống, không gây tổn kém chi phí cũng như sự trải nghiệm của khách hàng một cách trọn trù.

Để hoàn thành đồ án này, ngoài sự nỗ lực của bản thân, em xin chân thành gửi lời cảm ơn sâu sắc đến Trung tâm nghiên cứu và phát triển điện thoại di động Samsung tại Việt Nam – SVMC, cán bộ hướng dẫn tại Công ty cùng Ban lãnh đạo trường Đại học Công nghệ thông tin và Truyền thông Thái Nguyên và các thầy cô trong trường, đặc biệt là thầy **TS. Nguyễn Văn Núi** đã tận tình hướng dẫn, giúp đỡ và tạo điều kiện tốt nhất cho em kể từ khi bắt đầu nhận đề tài đến khi hoàn thành đề tài này.

Em xin chân thành cảm ơn!

Mục lục

LỜI MỞ ĐẦU.....	i
CHƯƠNG 1. GIỚI THIỆU ANDROID	1
1.1. Khái niệm về Android	1
1.1.1. Android khác với các hệ điều hành chạy trên thiết bị di động khác.....	2
1.1.2. Đặc tính mở của Android.....	2
1.2. Kiến trúc của Android	3
1.2.1. Android Platform.....	3
1.2.2. Tầng Linux Kernel	4
1.2.3. Tầng Native Libraries	4
1.2.4. Tầng Runtime	5
1.2.5. Tầng Application Framework	5
CHƯƠNG 2. MÔI TRƯỜNG VÀ CÔNG CỤ LẬP TRÌNH	7
2.1. Giới thiệu công cụ.....	7
2.2. Cài đặt Android Studio	7
2.3. Tạo ứng dụng đầu tiên	11
2.4. Thành phần quan trọng trong một Android Project.....	19
2.5. Chu kỳ sống của ứng dụng Android.....	20
2.5.1. Chu kỳ sống	20
2.5.2. Activity Stack.....	20
2.5.3. Các trạng thái của chu kỳ sống	21
2.5.4. Các hàm thực thi.....	22
2.6. Các Layout trong Android.....	23

2.6.1.	Frame Layout	23
2.6.2.	LinearLayout	24
2.6.3.	Relative Layout	26
2.6.4.	TableLayout	28
2.6.5.	GridLayout	29
2.7.	broadcast receiver	31
2.7.1.	Cách đăng ký Broadcast	32
2.7.2.	Custom BroadcastReceiver	34
2.7.3.	Local Broadcast Receiver	35
2.7.4.	Các lưu ý về bảo mật và cách xử lí	35
2.8.	Service	36
2.8.1.	Phân loại Service	37
2.8.2.	Độ ưu tiên các loại Service	38
2.8.3.	Các giá trị trả về trong onStartCommand()	39
2.8.4.	Các phương thức quan trọng trong vòng đời Service	40
2.8.5.	Running một Foreground Service	41
2.8.6.	Intent Service	42
CHƯƠNG 3. MÔ TẢ VÀ PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....		44
3.1.	Tổng quan về Samsung Knox	44
3.1.1.	Samsung Knox	44
3.1.2.	Knox SDK.....	45
3.2.	Mô tả và yêu cầu hệ thống	46
3.2.1.	<i>So sánh hệ thống cũ</i>	46

3.2.2. Mô tả hệ thống	47
3.2.3. Yêu cầu hệ thống	49
3.3. Phân tích thiết kế hệ thống.....	50
3.3.1. Biểu đồ use case tổng quát.....	50
3.3.2. Phân rã use case.....	51
3.3.3. Biểu đồ trạng thái	53
3.3.4. Biểu đồ trình tự	58
CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG	64
4.1. Giao diện bắt đầu khởi chạy ứng dụng	64
4.2. Giao diện cấp quyền cho ứng dụng	65
4.3. Giao diện đăng ký.....	66
4.4. Giao diện xác nhận mật khẩu	67
4.5. Giao diện quét thẻ NFC.....	68
4.6. Giao diện trang chủ	69
4.7. Giao diện thay đổi mật khẩu và thẻ NFC.....	70
4.8. Giao diện đặt thời gian phát video.....	71
4.9. Giao diện phát video.....	72
4.10. Giao diện cảnh báo trộm	73
4.11. Đánh giá thời gian sử dụng pin của ứng dụng	74
4.12. Đánh giá sử dụng ram của các thiết bị khác nhau.....	75
KẾT LUẬN.....	77
NHẬN XÉT.....	79
TÀI LIỆU THAM KHẢO.....	80

Danh mục hình ảnh

Hình 3. 1 Biểu đồ use case tổng quát.....	50
Hình 3. 2 Biểu đồ phân rã chức năng thay đổi mật khẩu.....	51
Hình 3. 3 Biểu đồ phân rã chức năng thay đổi thẻ NFC.....	51
Hình 3. 4 Biểu đồ phân rã chức năng khóa và mở khóa thiết bị.....	52
Hình 3. 5 Biểu đồ trạng thái “kích hoạt ứng dụng”	53
Hình 3. 6 Biểu đồ trạng thái “Đăng xuất và hủy kích hoạt”	54
Hình 3. 7 Biểu đồ trạng thái “Thay đổi mật khẩu”.....	54
Hình 3. 8 Biểu đồ trạng thái “Đặt thời gian phát video”	55
Hình 3. 9 Biểu đồ trạng thái “Phát video”	56
Hình 3. 10 Biểu đồ trạng thái “Khóa và cảnh báo”	56
Hình 3. 11 Biểu đồ trạng thái chức năng “Mở khóa thiết bị”	57
Hình 3. 12 Biểu đồ trình tự chức năng “Kích hoạt ứng dụng”	58
Hình 3. 13 Biểu đồ trình tự cho chức năng “Thay đổi thẻ NFC”.....	59
Hình 3. 14 Biểu đồ trình tự cho chức năng “Hủy kích hoạt”	60
Hình 3. 15 Biểu đồ trình tự cho chức năng “Phát video intro”	61
Hình 3. 16 Biểu đồ trình tự cho chức năng “khóa thiết bị”.	62
Hình 3. 17 Biểu đồ trình tự cho chức năng “Mở khóa thiết bị”.....	63
Hình 4. 1 Giao diện khởi chạy ứng dụng	64
Hình 4. 2 Giao diện xin cấp quyền Admin.....	65
Hình 4. 3 Giao diện đăng ký	66
Hình 4. 4 Giao diện xác nhận mật khẩu.....	67
Hình 4. 5 Giao diện đăng ký thẻ NFC.....	68
Hình 4. 6 Giao diện trang chủ.....	69
Hình 4. 7 Giao diện thay đổi mật khẩu	70

Hình 4. 8 Giao diện đặt thời gian phát video	71
Hình 4. 9 Giao diện phát video	72
Hình 4. 10 Giao diện cảnh báo trộm	73
Hình 4. 11 Biểu đồ kiểm thử thời gian sử dụng pin	74
Hình 4. 12 Biểu đồ sử dụng ram trên các thiết bị khác nhau	75

CHƯƠNG 1. GIỚI THIỆU ANDROID

Như chúng ta biết, hiện tại đã có hơn nửa nhân loại sử dụng máy di động để thoại và giao tiếp qua các mạng không dây. Con số 3 tỷ người này sẽ còn tăng lên và máy di động càng ngày càng "thông minh" với nhiều chức năng và dịch vụ rất hấp dẫn, cho nên thị trường máy di động thông minh sẽ vượt xa máy vi tính trong một tương lai rất gần... Vì thế việc lập trình trên thiết bị di động ngày càng phổ biến và phát triển rất mạnh mẽ. Từ nền tảng mã nguồn mở, Google đã cho ra mắt Android chạy trên các thiết bị di động. Android có rất nhiều công cụ và dụng cụ miễn phí để nghiên cứu và phát triển phần mềm trên nền tảng của nó. Tài liệu này sẽ giúp chúng ta tìm hiểu về Android và cách viết một ứng dụng trên nền tảng này.

1.1. Khái niệm về Android

Trước hết Android là nền tảng phần mềm dựa trên mã nguồn mở của Linux OS (Kernel 2.6) cho máy di động và những phần mềm trung gian (middleware) để hỗ trợ các ứng dụng mà người sử dụng cần đến. Một cách định nghĩa không quá chuyên môn thì có thể coi Android là một nền tảng mở cho thiết bị di động của Google (gồm hệ điều hành, middleware và một số ứng dụng cơ bản). Android sẽ đương đầu với một số hệ điều hành (viết tắt là HDH) dành cho thiết bị di động khác đang hâm nóng thị trường như Windows Mobile, OS X (iPhone).

Có thể nói một cách nôm na rằng Android là một HDH chạy trên thiết bị di động, Cũng giống như Windows, Linux hay Mac chạy trên máy vi tính vậy.

1.1.1. Android khác với các hệ điều hành chạy trên thiết bị di động khác

Android đã thu hút được sự chú ý của giới công nghệ khác toàn cầu khi đưa con của Google sử dụng giấy phép mã nguồn mở. Đó là một sản phẩm kết tinh từ ý tưởng của Khối Liên minh thiết bị cầm tay do Google dẫn đầu, gồm 34 thành viên với các công ty hàng đầu về công nghệ và di động toàn cầu như Qualcomm, Intel, Motorola, và LG Electronics, các nhà mạng như T-Mobile, Sprint Nextel à China Mobile.

Các nhà phát triển có thể sử dụng miễn phí bộ Kit Android Software Development để xây dựng các ứng dụng của mình.

1.1.2. Đặc tính mở của Android

Android được xây dựng để cho phép các nhà phát triển tạo ra các ứng dụng di động hấp dẫn, tận dụng tất cả tính năng một chiếc điện thoại đã cung cấp.

Android được xây dựng để mở. Ví dụ, một ứng dụng có thể gọi bất kỳ chức năng của lõi điện thoại như thực hiện cuộc gọi gửi tin nhắn văn bản, chụp ảnh... Cho phép các nhà phát triển tạo ra nhiều ứng dụng phong phú hơn cho người dùng (điều này hiện chưa có trên Windows Phone 7 của Microsoft). Android được xây dựng trên mã nguồn mở của Linux Kernel. Hơn nữa, nó sử dụng một máy ảo tùy chỉnh được thiết kế để tối ưu hóa bộ nhớ và tài nguyên phần cứng trong môi trường di động.

Android cung cấp truy cập đến một loạt các thư viện công cụ hữu ích và có thể được sử dụng để xây dựng các ứng dụng phong phú. Ví dụ, Android cho phép các thiết bị giao tiếp với nhau tạo điều kiện cho đồng đẳng rich-to-peer trong ứng dụng xã hội. Ngoài ra, Android bao gồm một tập hợp đầy đủ công cụ đã được xây dựng công phu,

với việc cung cấp nền tảng phát triển, với năng suất cao và cái nhìn sâu vào các ứng dụng.

1.2. Kiến trúc của Android

“Understanding Android” là cách mà ta tiếp cận lập trình Android và thấu hiểu kiến trúc hệ thống của nó. Chúng ta có thể không cần hiểu rõ cấu trúc của một hệ điều hành nhưng chúng ta vẫn có thể lập trình một ứng dụng trên hệ điều hành đó, đây là một điều mà nhà sản xuất muốn khi release SDK với một framework có sẵn của họ. Như chúng ta biết điều này cũng có mặt tốt và xấu. Framework là một tầng cao cấp dành cho lập trình viên, nó đều có giới hạn của nó, chúng ta chỉ có thể lập trình những ứng dụng phổ biến nhưng không nên tiến tới những ứng dụng cao cấp đi sâu vào hệ thống của hệ điều hành.

1.2.1. Android Platform

Bao gồm hệ điều hành Android đầy đủ tính năng, các ứng dụng và các tầng trung gian để developer có thể mở rộng, tùy chỉnh hoặc thêm vào các component của họ.

Có 4 tầng cơ bản trong hệ điều hành Android:

- Application Framework
- Android Runtime
- Native Libraries
- Linux Kernel

Mỗi tầng làm việc đều nhờ sự giúp đỡ của tầng bên dưới.

1.2.2. Tầng Linux Kernel

Đây là tầng nhân của hệ điều hành Android, mọi việc xử lý của hệ thống đều phải thông qua tầng này. Linux Kernel cung cấp các trình điều khiển thiết bị phần cứng (Driver) như camera, USB, Wifi, Bluetooth, Display, Power Management...

Android dựa trên Linux phiên bản 2.6 lựa chọn tính năng cốt lõi như bảo mật, quản lý bộ nhớ, quản lý tiến trình, mạng stack và các trình điều khiển phần cứng. Kernel hoạt động như một lớp trừu tượng giữa phần cứng và phần mềm còn lại của hệ thống

1.2.3. Tầng Native Libraries

- System C library: Có nguồn gốc từ hệ thống thư viện chuẩn C (libc), điều chỉnh các thiết bị nhúng trên Linux.
- Media Libraries: Mở rộng từ PacketVideo's OpenCORE. Là thư viện hỗ trợ playback và recording của nhiều định dạng video và image phổ biến như MPEG4, H.264, MP3, AAC, JPG và PNG...
- Surface Manager: Quản lý việc hiển thị và kết hợp đồ họa 2D và 3D.
- LibWebCore: Android dung lại Webkit Engine cho việc render trình duyệt mặc định của hệ điều hành Android browser và cho định dạng nhúng như HTML.
- SGL: 2D engine
- 3D libraries: Thư viện 3D dựa trên OpenGL ES 1.0 API, có nâng cấp tăng tốc "hardware 3D acceleration"
- FreeType: Render bitmap và vector font

- SQLite: Quản lý database của ứng dụng

1.2.4. Tầng Runtime

Một ứng dụng Android chạy trên một process riêng của Dalvik VM (Máy ảo). Dalvik được viết để chạy nhiều máy ảo cùng một lúc một cách hiệu quả trên cùng một thiết bị.

Máy ảo Dalvik thực thi các file mang định dạng `**.dex` (Dalvik Executable), định dạng này đã được tối ưu hóa chỉ chiếm một vùng nhớ nhỏ vừa đủ và nhỏ nhất có thể. Máy ảo chạy các class (đã được compile trước đó bởi 1 trình biên dịch ngôn ngữ Java), sở dĩ máy ảo chạy được các class này là nhờ chương trình DEX tool đã convert các class sang định dạng `**.dex`.

1.2.5. Tầng Application Framework

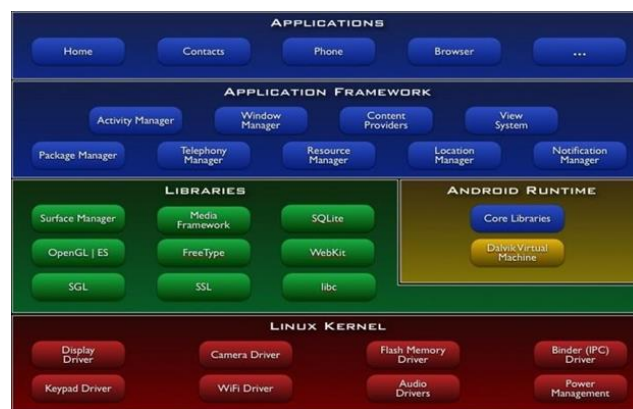
Đây là tầng mà Google xây dựng cho phép các Developer để phát triển các ứng dụng của họ trên Android, chỉ bằng cách gọi các API có sẵn mà Google đã viết để sử dụng các tính năng của phần cứng mà không cần hiểu cấu trúc bên dưới.

Bằng cách cung cấp một nền tảng phát triển mở, Android cho các nhà phát triển khả năng xây dựng các ứng dụng cực kỳ phong phú và sáng tạo. Nhà phát triển được tự do tận dụng các thiết bị phần cứng, thông tin địa điểm truy cập, các dịch vụ chạy nền, thiết lập hệ thống báo thức, thêm các thông báo để các thanh trạng thái và nhiều hơn nữa.

Tất cả các ứng dụng thường gồm một bộ các dịch vụ và hệ thống cơ bản sau:

- View UI: dùng để xây dựng layout của ứng dụng bao gồm: listview, text field, button, dialog, from...
- Content Providers: cho phép các ứng dụng có thể truy cập dữ liệu từ các ứng dụng khác (như ứng dụng của ta có thể lấy thông tin Contacts của điện thoại Android), hoặc để chia sẻ dữ liệu của riêng ứng dụng.
- Resource Manager: cung cấp cách thức truy cập đến non-code resources như các asset, graphic, image, music, video...
- Notification Manager: cho phép tất cả các ứng dụng hiển thị thông báo của mình trên hệ điều hành.
- Activity Manager: quản lý vòng đời của các ứng dụng.

Ở góc nhìn của người dùng ta có thêm tầng Application (là tầng ứng dụng do chúng ta viết). Sau đây là sơ đồ tổng quát như hình dưới.



Hình 1. 1 Kiến trúc Android

CHƯƠNG 2. MÔI TRƯỜNG VÀ CÔNG CỤ LẬP TRÌNH

2.1. Giới thiệu công cụ

Trong chương này sẽ giới thiệu các công cụ lập trình cho Android (Android Development Tools). Chúng ta sẽ dần làm quen với Android Studio.

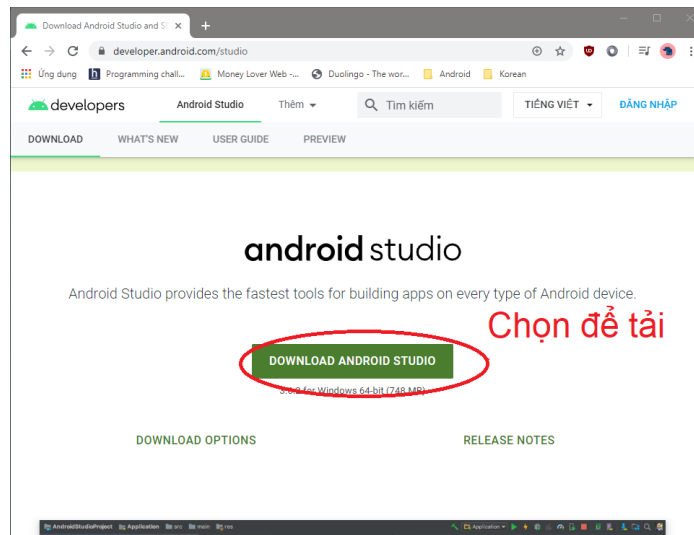
Android Studio là môi trường phát triển tích hợp IDE chính thức dành cho phát triển nền tảng Android.

Nó được ra mắt vào ngày 16 tháng 5 năm 2013, được phát hành miễn phí theo giấy phép Apache Licence 2.0.

Android Studio được phát triển dựa trên phần mềm 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, Mac OS và Linux. Là IDE chính thức của Google để phát triển ứng dụng Android gốc để thay thế cho Android Development Tools dựa trên Eclipse.

2.2. Cài đặt Android Studio

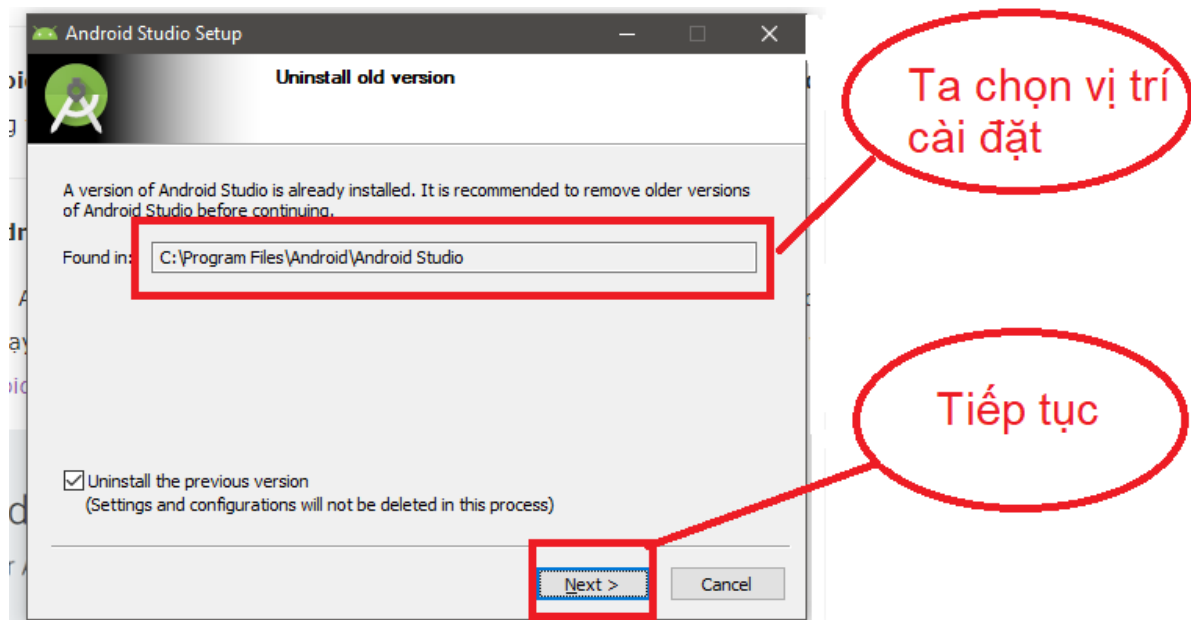
Bước 1: Chúng ta truy cập vào trang chủ của Android Studio để tải bản mới nhất <https://developer.android.com/studio>.



Hình 2. 1 Trang chủ tải Android Studio

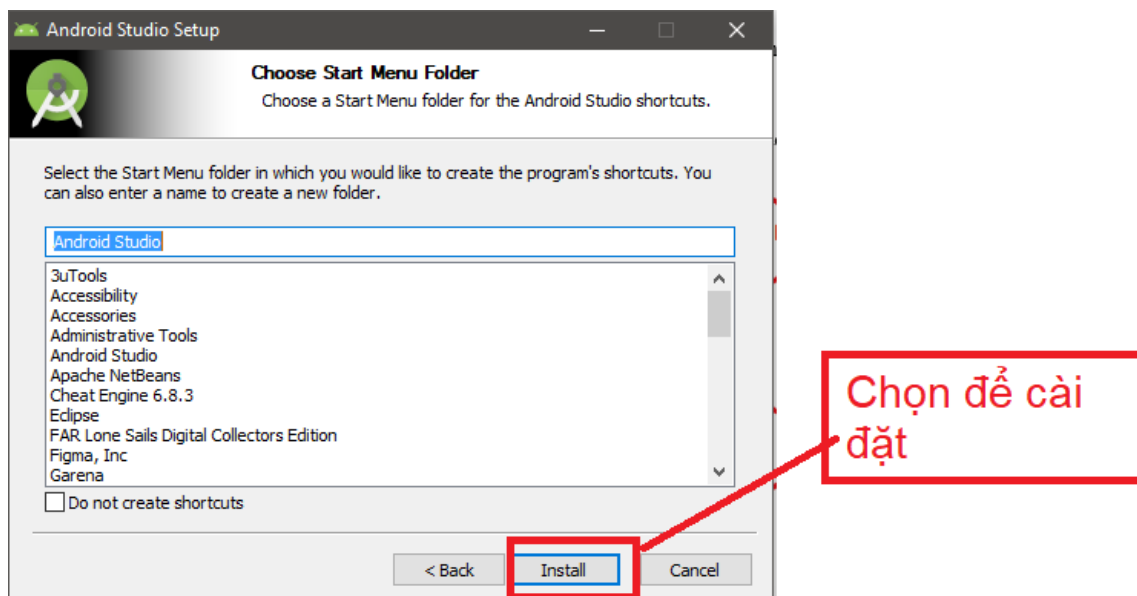
Bước 2: Tiến hành cài đặt trên Windows. Sau khi tải xuống sau tại thực hiện việc chạy file vừa tải xuống.

Ta có thể chọn vị trí lưu cài đặt Android Studio. Sau đó ấn **Next** để tiếp tục.



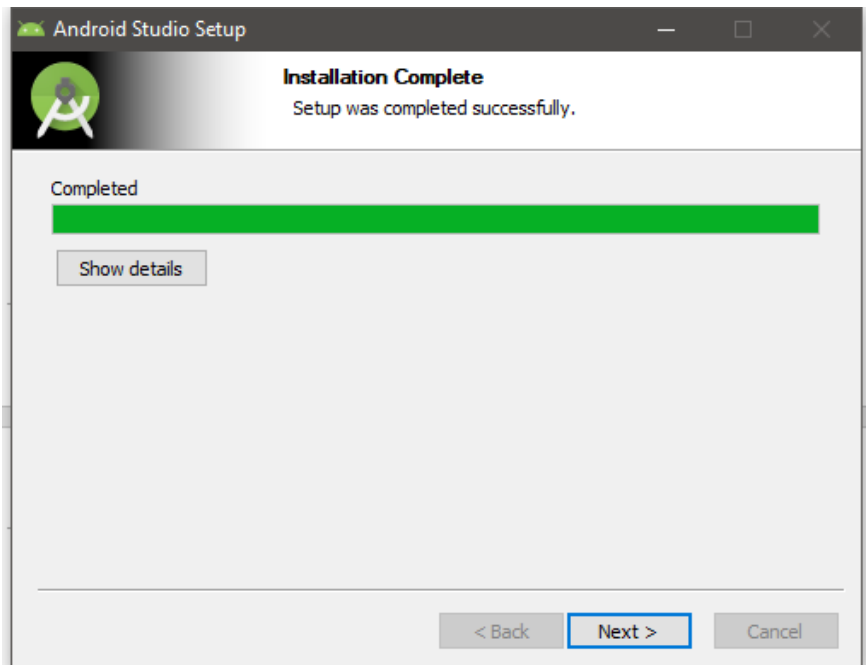
Hình 2. 2 Chọn thư mục cài đặt Android Studio

Bước 3: Sau đó ta chọn Install để cài đặt. Và đợi Android Studio tự cài đặt.



Hình 2. 3 Chọn Install để cài đặt Android Studio

Bước 4: Hoàn thành việc cài đặt. Ta chọn NEXT để hoàn thành khi Android Studio đã cài đặt xong.



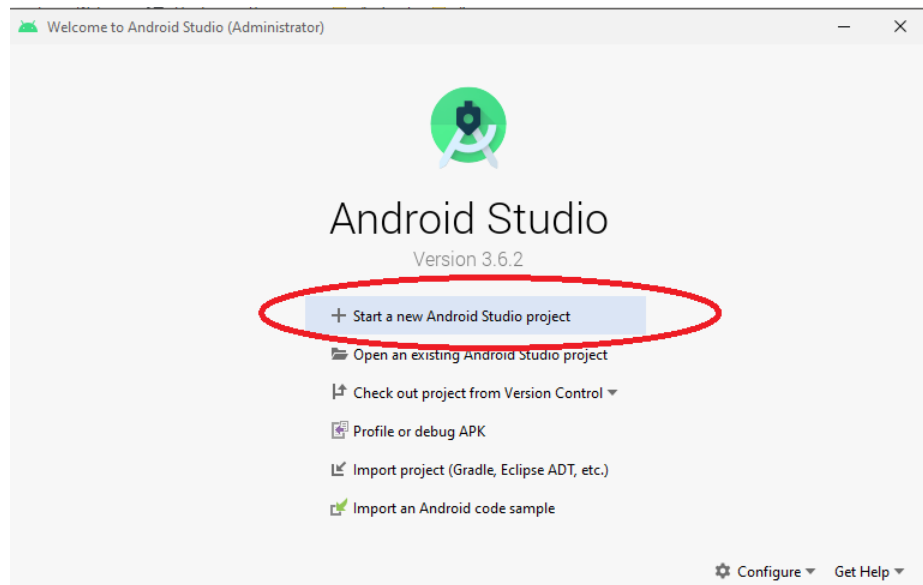
Hình 2. 4 Hoàn thành việc cài đặt Android Studio



Hình 2. 5 Đã hoàn thành việc cài đặt Android Studio

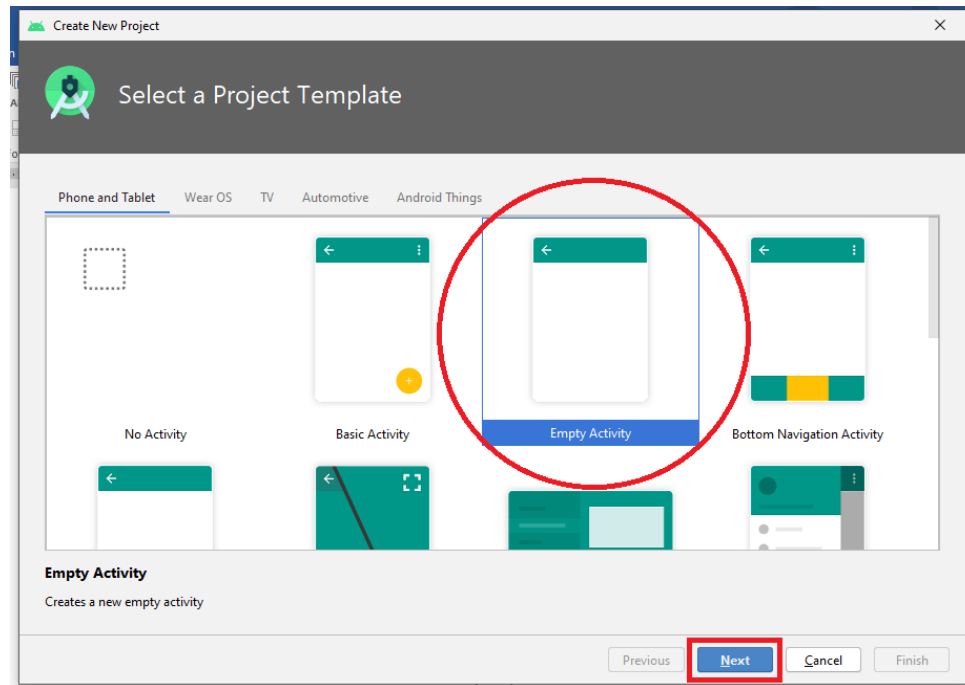
2.3. Tạo ứng dụng đầu tiên

Bước 1: Sau khi mở ứng dụng. Ta thấy giao diện bắt đầu của Android Studio. Ta chọn “Start a new Android Studio project”. Để bắt đầu tạo ứng dụng.



Hình 2. 6 Bắt đầu tạo project mới

Bước 2 : Ta chọn Empty Activity. Sau đó chọn Next.

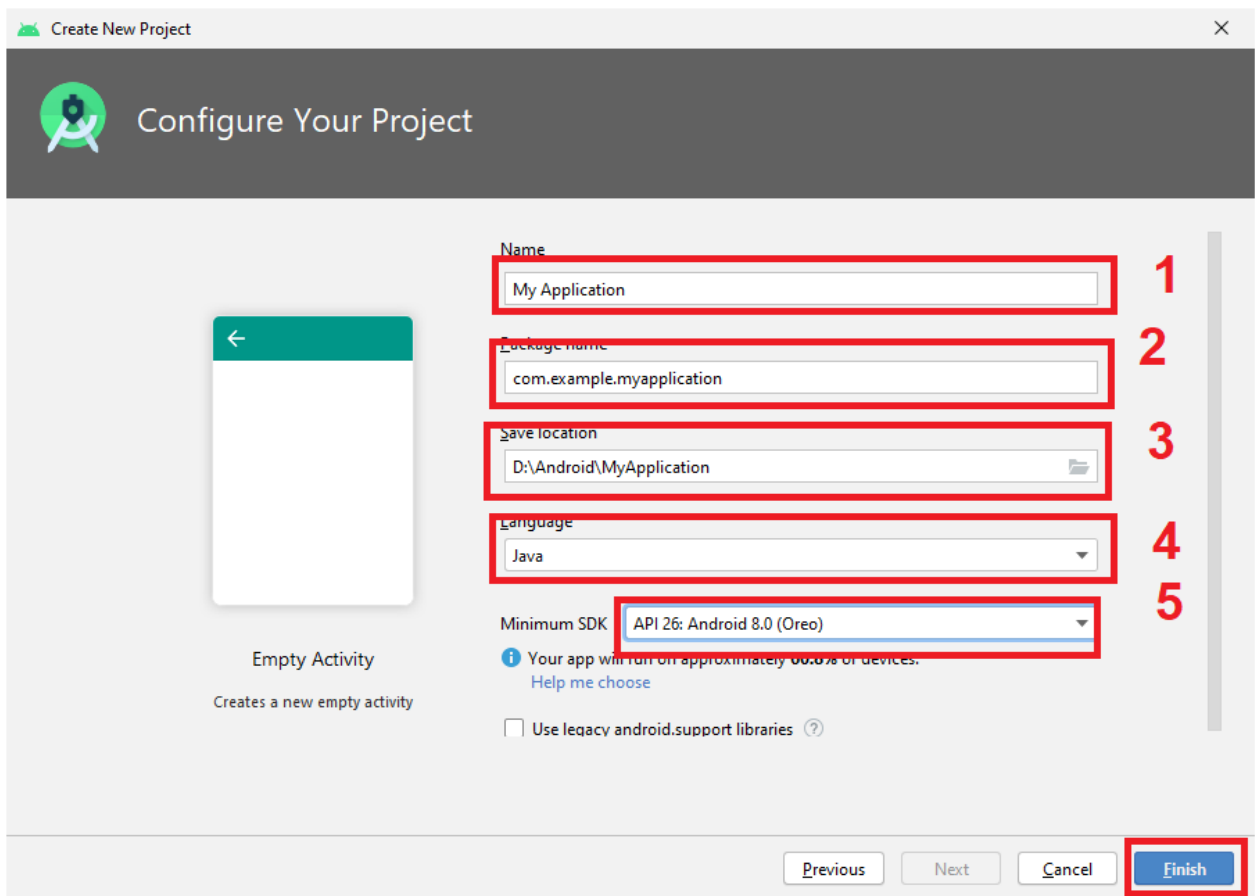


Hình 2. 7 Bắt đầu tạo một Empty Activity

Bước 3: Như hình minh họa dưới.

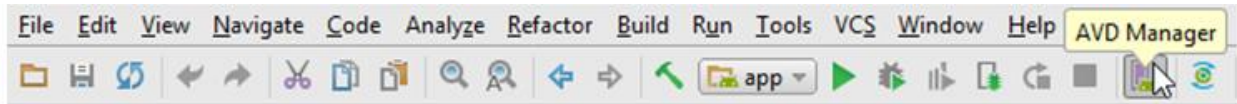
1. Là tên của dự án
2. Tên của gói dự án
3. Nơi lưu trữ dự án
4. Là ngôn ngữ, hiện tại thì sẽ có 2 ngôn ngữ lập trình cho Android là Java và Kotlin
5. Là phiên bản thấp nhất sử dụng SDK.

Sau khi chọn xong chúng ta chọn Next để tiếp tục.



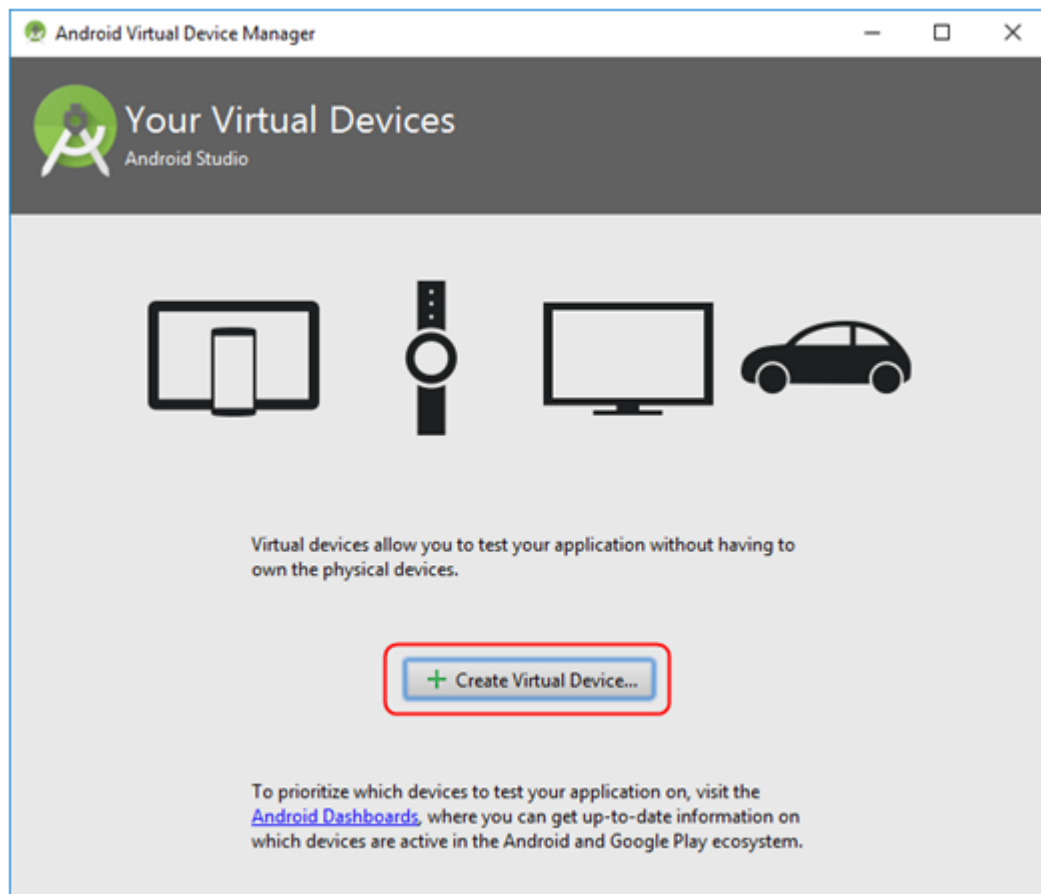
Hình 2. 8 Mô tả việc chọn phiên bản, ngôn ngữ lập trình và đặt tên dự án

Bước 4: Tạo máy ảo. Ta chọn vào biểu tượng AVD Manager trên thanh Toolbar.



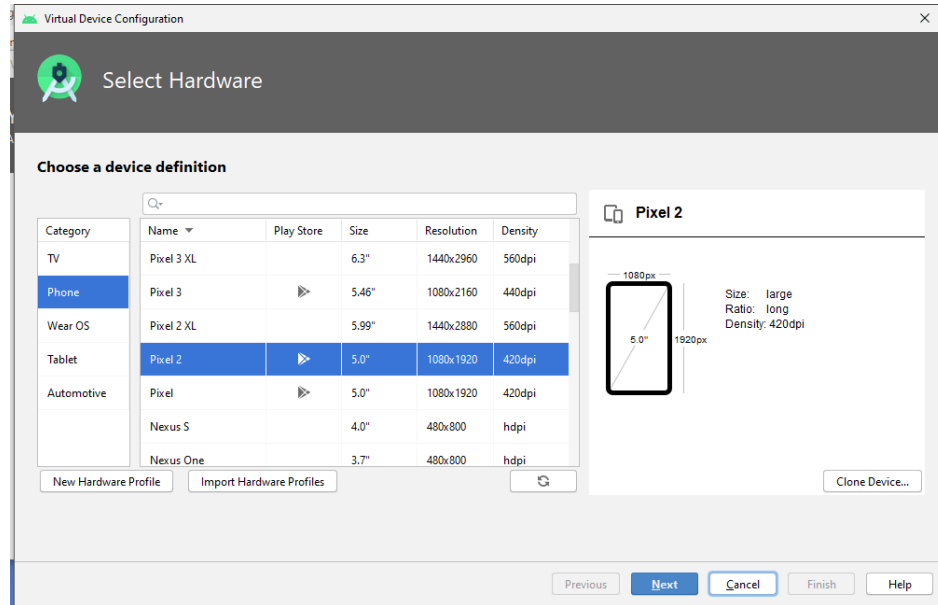
Hình 2. 9 Chọn Manager AVD

Bước 5: Chọn Create Virtual Device. Để bắt đầu tạo máy ảo.



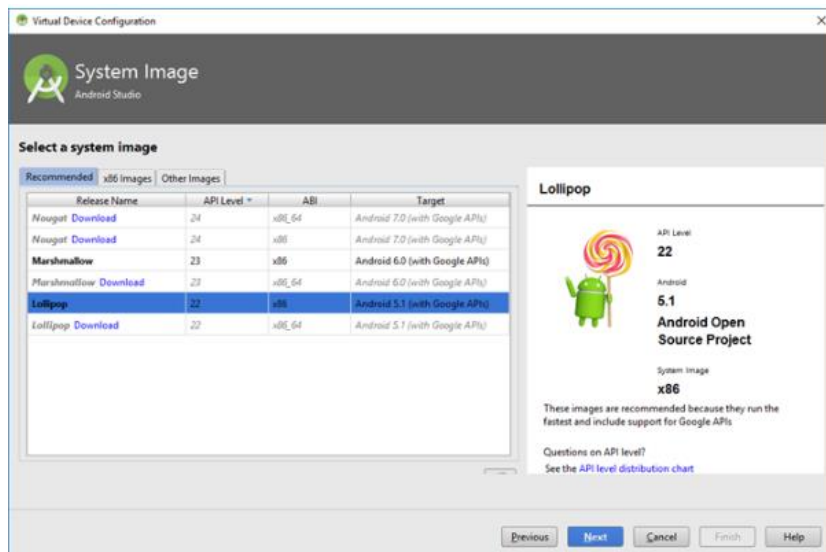
Hình 2. 10 Create Virtual Device

Bước 6: Chọn máy ảo muốn tạo. Sau đó ta ấn “Next”.




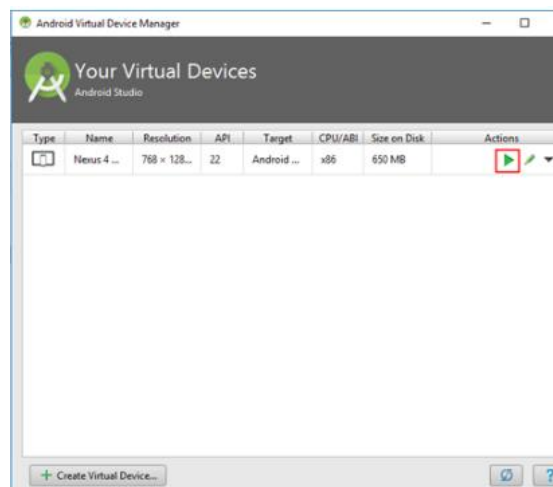
Hình 2. 11 Chọn máy ảo

Bước 7: Chọn hệ điều hành cho máy ảo.



Hình 2. 12 Chọn hệ điều hành cho máy ảo.


Bước 8: Chọn vào  để bắt đầu khởi động máy ảo.



Hình 2. 13 Danh sách máy ảo



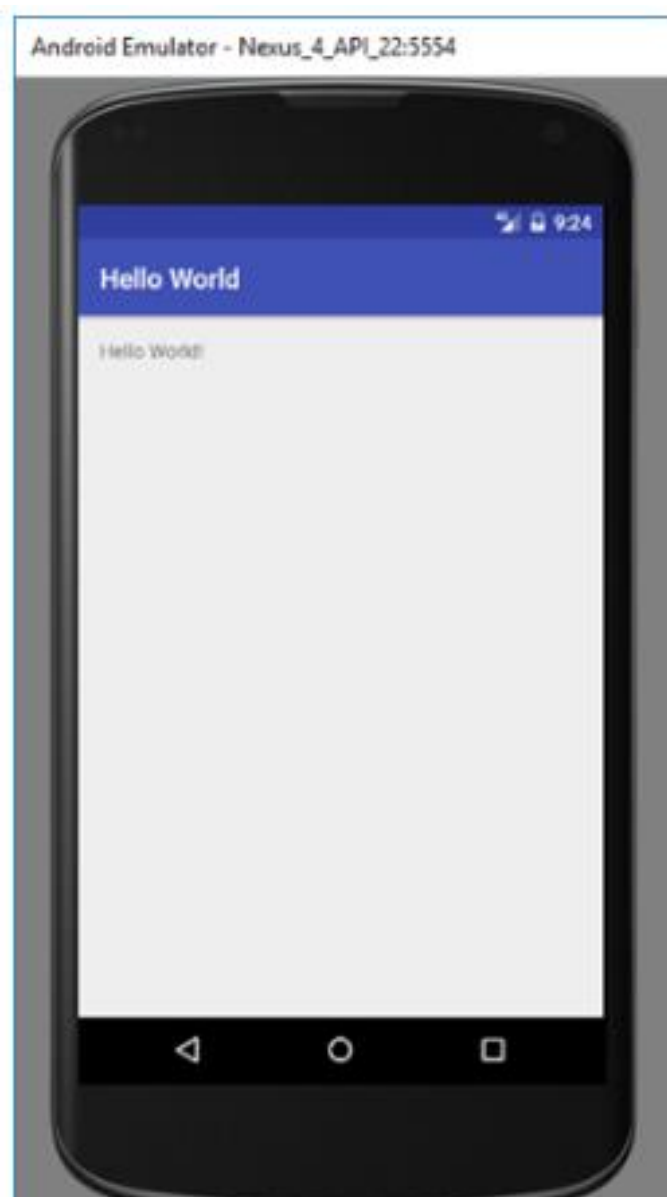
Hình 2. 14 Máy ảo đã khởi động xong

Bước 9: Giờ ta có thể tiến hành Build app bằng cách chọn vào biểu tượng  để bắt đầu Build.



Hình 2. 15 Toolbar Build

Sau khi Build và cài đặt xong ta sẽ có kết quả như hình bên dưới.



Hình 2. 16 Chương trình đầu tiên

2.4. Thành phần quan trọng trong một Android Project

- Activity: Đây là một lớp khởi tạo giao diện ứng dụng nội bộ trên Android.

- Service: Cung cấp các dịch vụ liên quan đến client/service. Một Service sẽ chạy ngầm bên dưới.
- Broadcast receiver: đây là một ứng dụng chạy ngầm dùng để đọc và nhận thông tin trên UI, ví như cập nhật pin, ngày giờ...
- Content Provider: Cung cấp chức năng truy vấn dữ liệu giữa các ứng dụng của Android
- Intent: Là nền tảng để truyền tải các thông báo. Intent được sử dụng để gửi các thông báo đi nhằm khởi tạo 1 Activity hay Service để thực hiện công việc mà chúng mong muốn.

2.5. Chu kỳ sống của ứng dụng Android

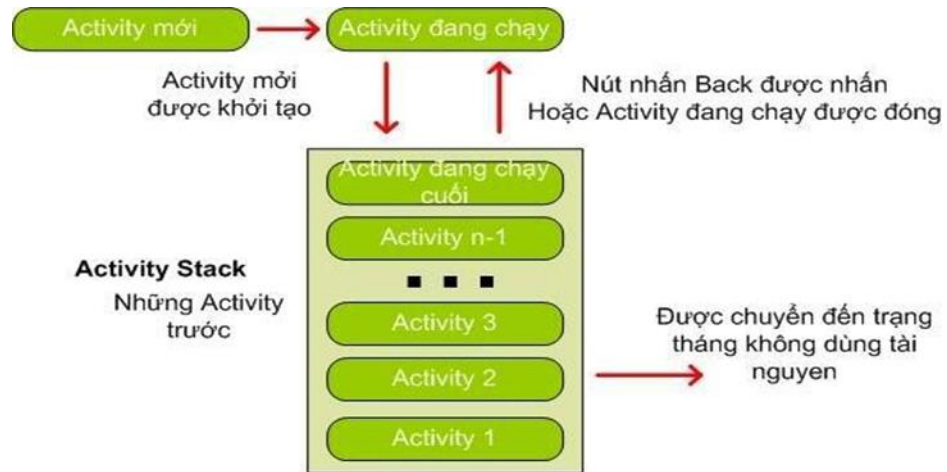
2.5.1. Chu kỳ sống

Các thành phần ứng dụng có một chu kỳ sống từ lúc khởi tạo và đến thời điểm kết thúc.

2.5.2. Activity Stack

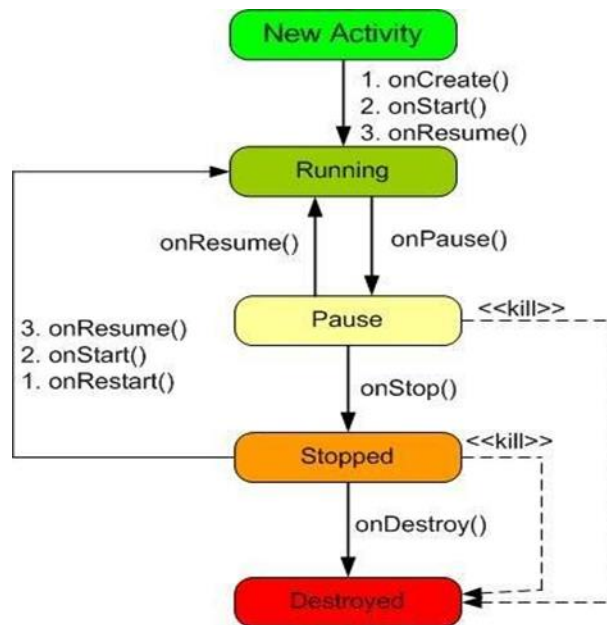
Bên trong hệ thống các Activity được quản lý như một activity stack. Khi một activity mới được chạy nó được đặt ở đỉnh của stack và trở thành activity đang chạy. Activity trước sẽ ở bên dưới activity mới và sẽ không thấy trong suốt quá trình activity mới tồn tại

Nếu người dùng ấn nút “Back” thì activity kết tiếp của stack sẽ di chuyển lên trên và trở thành active.



Hình 2. 17 Activity Stack

2.5.3. Các trạng thái của chu kỳ sống



Hình 2. 18 Chu kỳ sống của Activity

Một Activity chủ yếu có 4 chu kỳ chính là:

- Active hoặc running: Khi Active là được chạy trên màn hình. Activity này tập trung vào những thao tác của người dùng trên ứng dụng.
- Paused: là Activity được tạm dừng khi mất focus nhưng người dùng vẫn trông thấy.
- Stopped: Nếu nó hoàn toàn bị bao phủ bởi một Activity khác. Nó vẫn có trạng thái và thông tin thành viên trong nó. Người dùng không thấy được nó và thường bị loại bỏ trong trường hợp hệ thống cần vùng nhớ cho tác vụ khác.
- Killed: Khi hệ thống thiếu vùng nhớ nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái Stop hay Pause cũng có thể bị giải phóng.

2.5.4. Các hàm thực thi

- onCreate(...): hàm này được gọi khi lớp Activity được khởi tạo, dùng để thiết lập giao diện ứng dụng và thực thi những thao tác cơ bản.
- onStart(): hàm này được gọi khi lớp ứng dụng xuất hiện trên màn hình.
- onResume(): hàm được gọi ngay sau onStart hoặc khi người dùng focus ứng dụng, hàm này sẽ đưa ứng dụng lên top màn hình.
- onPause(): hàm được gọi khi hệ thống đang focus đến 1 activity trước đó.
- onStop(): hàm được gọi khi một activity khác được khởi động và focus.
- onRestart(): được gọi khi ứng dụng chuyển sang onStop(), nhưng muốn khởi động lại bằng onStart().

2.6. Các Layout trong Android

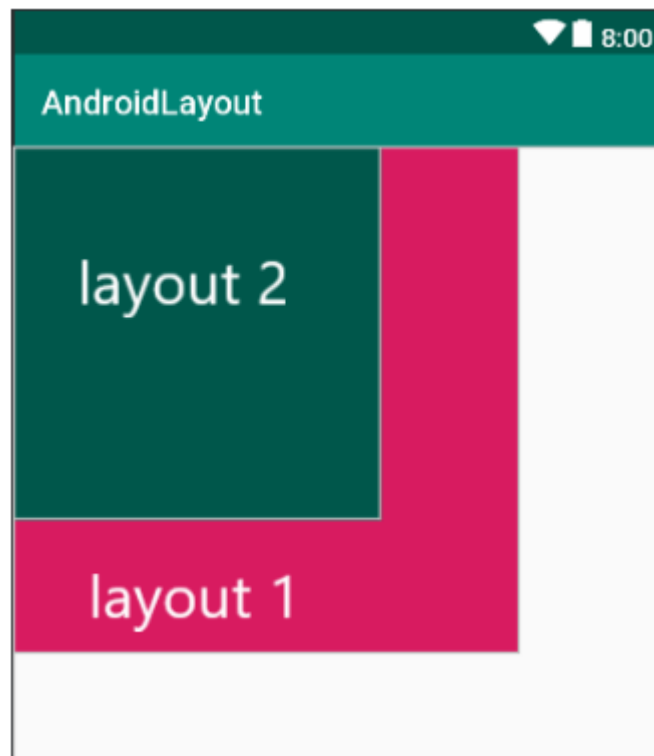
2.6.1. Frame Layout

Frame Layout là một dạng Layout cơ bản nhất khi gắn các View lên layout này thì nó sẽ luôn giữ các View này ở phía góc trái màn hình và không cho chúng thay đổi vị trí của chúng. Các view sau đưa vào sau sẽ đè lên view ở trước trừ khi bạn thiết lập “*transparent*” cho view sau đó.

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity" >

  <TextView
    android:id="@+id/textView"
    android:layout_width="300dp"
    android:layout_height="300dp"
    android:background="@color/colorAccent"
    android:text="TextView" />

  <TextView
    android:id="@+id/textView2"
    android:layout_width="218dp"
    android:layout_height="221dp"
    android:background="@color/colorPrimaryDark" />
</FrameLayout>
```



Hình 2. 19 Minh họa *FrameLayout*

2.6.2. *LinearLayout*

LinearLayout là loại layout thường được sử dụng. Được bố trí theo dạng khối và không đè lên nhau. *LinearLayout* được bố trí theo hai bố cục “*Vertical Orientation*” bố trí theo chiều dọc và “*Horizontal Orientation*” bố trí theo chiều ngang.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >

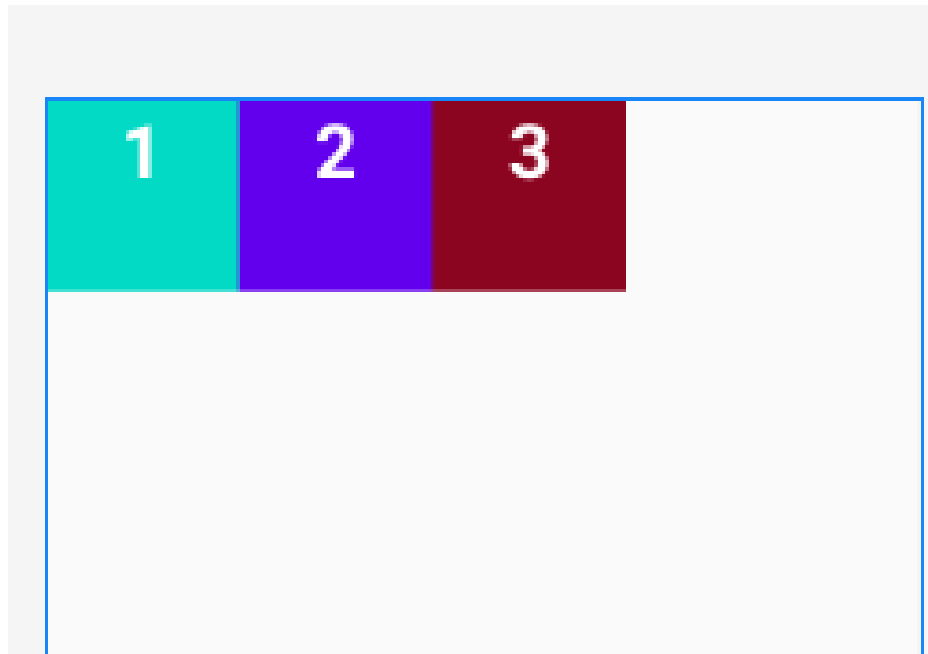
    <TextView
```

```

        android:layout_width="80dp"
        android:layout_height="80dp"
        android:text="1"
        android:textColor="#fff"
        android:textSize="15pt"
        android:textAlignment="center"
        android:textStyle="bold"
        android:background="@color/colorAccent" />

<TextView
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:text="2"
    android:textColor="#fff"
    android:textSize="15pt"
    android:textAlignment="center"
    android:textStyle="bold"
    android:background="@color/colorPrimary" />
<TextView
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:text="3"
    android:textColor="#fff"
    android:textSize="15pt"
    android:textAlignment="center"
    android:textStyle="bold"
    android:background="#8c0520" />
</LinearLayout>

```

Hình 2. 20 Minh họa LinearLayout

2.6.3. Relative Layout

RelativeLayout là một loại Layout mà trong đó vị trí của mỗi View con sẽ được xác định so với view khác hoặc so với thành phần cha của chúng thông qua ID.

Sẽ có 4 tính cơ bản để sử dụng

- Layout_below: Nằm bên dưới.
- Layout_above: Nằm bên trên
- Layout_toRightOf: Nằm bên phải.
- Layout_toLeftOf: nằm bên trái.



Hình 2. 21 Minh họa RelativeLayout

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:id="@+id/parent"
tools:context=".MainActivity">
    <TextView
        android:background="@color/colorAccent"
        android:textAlignment="center"
        android:textSize="50sp"
        android:text="1"
        android:id="@+id/tview1"
        android:layout_width="100dp"
        android:layout_height="100dp"
        tools:ignore="NotSibling" />
    <TextView
        android:layout_below="@+id/tview1"
        android:background="@color/colorPrimary"
        android:textAlignment="center"
```

```

        android:textSize="50sp"
        android:text="2"
        android:id="@+id/tview2"
        android:layout_width="100dp"
        android:layout_height="100dp"
        tools:ignore="NotSibling" />
    <TextView
        android:layout_above=""
        android:layout_toRightOf="@+id/tview1"
        android:background="@android:color/holo_red_dark"
        android:textAlignment="center"
        android:textSize="50sp"
        android:text="3"
        android:id="@+id/tview3"
        android:layout_width="100dp"
        android:layout_height="100dp"
        tools:ignore="NotSibling" />
</RelativeLayout>

```

2.6.4. *TableLayout*

TableLayout sẽ sắp xếp các View con bên trong thành dạng bảng. Mỗi hàng là một đối tượng View TableRow. Bên trong TableRow chứa các View con, mỗi View con này nằm ở một ô.

```

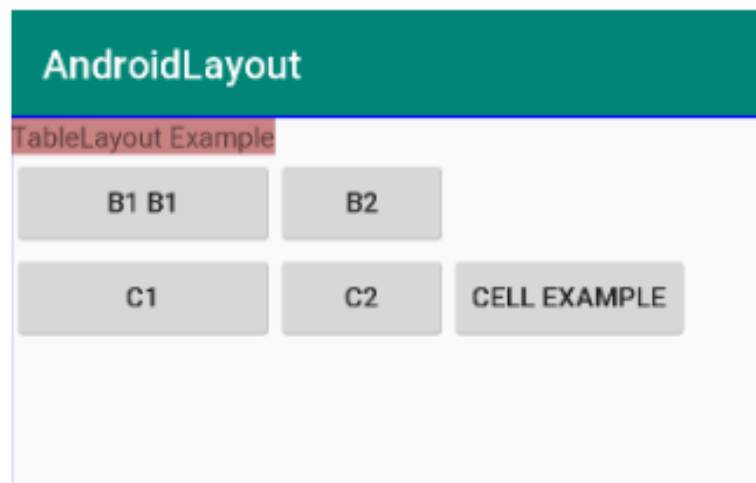
<TableLayout android:layout_height="match_parent"
    android:layout_width="match_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <TableRow>
        <TextView
            android:text="TableLayout Example"
            android:background="#c98282"
            android:gravity="center"/>
    </TableRow>
</TableLayout>

```

```

</TableRow>
<TableRow>
    <Button android:text="B1 B1" />
    <Button android:text="B2"/>
</TableRow>
<TableRow>
    <Button android:text="C1" />
    <Button android:text="C2" />
    <Button android:text="Cell example" />
</TableRow>
</TableLayout>

```



Hình 2. 22 Minh họa *TableLayout*

2.6.5. *GridLayout*

GridLayout là một dạng lưới và ta có thể chia các cột và dòng cho các lưới đó, các view sẽ được đặt vào các ô trong các lưới này.

```

<GridLayout
    xmlns:android1="http://schemas.android.com/apk/res/android"

```

```

        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:columnCount="2"
        android:rowCount="2">
        <TextView
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:text="1"
            android:textColor="#fff"
            android:textSize="15pt"
            android:textAlignment="center"
            android:background="@color/colorAccent" />
        <TextView
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:text="2"
            android:textColor="#fff"
            android:textSize="15pt"
            android:textAlignment="center"
            android:background="@color/colorPrimary" />
        <TextView
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:text="3"
            android:textColor="#fff"
            android:textSize="15pt"
            android:textAlignment="center"
            android:background="#8c0520" />
        <TextView
            android:layout_width="80dp"
            android:layout_height="80dp"
            android:background="#efcd21"
            android:text="4"
            android:textAlignment="center"
            android:textColor="#fff"
            android:textSize="15pt" />
    </GridLayout>

```



Hình 2. 23 Minh họa GridLayout

GridLayout được quy định bằng hai thuộc tính

`android:columnCount="2" // số hàng`

`android:rowCount="2" // số cột`

2.7. broadcast receiver

Broadcast Receiver là một trong 4 component lớn trong Android, với mục đích là lắng nghe các sự kiện, trạng thái của hệ thống phát ra thông qua Intent nhờ đó mà các lập trình viên có thể xử lý được các sự kiện hệ thống ở bên trong ứng dụng của mình.

Broadcast Receiver có thể hoạt động được cả khi ứng dụng bị tắt đi, nghĩa là ở background chính vì vậy nó thường được sử dụng với service.

2.7.1. Cách đăng ký Broadcast

- Đăng ký trong file AndroidManifest

Khai báo Broadcast Receiver trong Manifest và khi lần đầu khởi chạy ứng dụng thì nó sẽ đăng ký Broadcast.

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

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

    <receiver android:name=".Broadcast">
        <intent-filter>
            <action android:name="android.intent.action.AIRPLANE_MODE"/>
        </intent-filter>
    </receiver>
</application>
```

Hình 2. 24 Đăng ký broadcast trong manifest

Thực hiện đăng ký lắng nghe chúng ta sẽ xử lý kết quả trả về bằng cách tạo một class mới extends lại BroadcastReceiver.

```
public class Broadcast extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        Log.d(Broadcast.class.getSimpleName(), "Air Plane mode");
    }
}
```

Hình 2. 25 Lắng nghe Broadcast

- Đăng ký trong file Java

Đây là cách thông dụng nhất vì đa số trong các ứng dụng chúng ta chỉ lắng nghe trong phạm vi ứng dụng của mình, khi kết thúc thì chúng ta cũng ngừng lắng nghe luôn. Chúng ta cũng làm như ở cách đăng ký ở trong file Manifest chỉ khác biệt ở trong cách đăng ký.

```
public class MainActivity extends AppCompatActivity {

    private Broadcast broadcast;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        broadcast = new Broadcast();
        IntentFilter filter =
            new IntentFilter("android.intent.action.AIRPLANE_MODE");
        registerReceiver(broadcast, filter);
    }

    @Override
    protected void onStop() {
        super.onStop();
        unregisterReceiver(broadcast);
    }
}
```

Hình 2. 26 Hình code đăng ký trong file java

2.7.2. Custom BroadcastReceiver

Thường được sử dụng để truyền thông điệp trong và ngoài ứng dụng nhưng không phải là sự thay đổi từ hệ thống mà là những thông điệp mà lập trình viên muốn truyền đi.

Tạo receiver nhưng ở phần action thay vì sử dụng action có sẵn của hệ thống thì ta tự tạo 1 action.

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

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

    <receiver android:name=".Broadcast">
        <intent-filter>
            <action android:name="test.Broadcast"/>
        </intent-filter>
    </receiver>

    <activity android:name=".TestBroadcastActivity">
    </activity>
</application>
```

Hình 2. 27 Tạo receiver ở phần action

2.7.3. Local Broadcast Receiver

Không phải lo lắng về việc rò rỉ dữ liệu vì dữ liệu chỉ được gói gọn trong project của bạn mà thôi.

Không phải lo lắng về các lỗ hổng bảo mật vì nó không send broadcast với ứng dụng khác.

Nó hiệu quả hơn việc gửi một Broadcast qua hệ thống.

2.7.4. Các lưu ý về bảo mật và cách xử lý

- Không muốn sendBroadcast cho các đối tượng ở bên ngoài ứng dụng của bạn thì có thể sử dụng LocalBroadcast vì LocalBroadcastManager hiệu quả hơn (không cần giao tiếp liên tục) và cho phép tránh suy nghĩ về bất kỳ vấn đề bảo mật nào liên quan đến các ứng dụng khác có thể nhận hoặc gửi các broadcast.
- Không phát thông tin nhạy cảm bằng cách sử dụng intent không tường minh. Thông tin có thể được đọc bởi bất kỳ ứng dụng nào đăng ký để nhận broadcast. Có ba cách để kiểm soát nhận broadcast: set các quyền cho Broadcast; Set package mà bạn muốn gửi Broadcast đến quá setPackage(String); Sử dụng Local Broadcast.
- Khi đăng ký nhận Broadcast thì bất cứ ứng dụng nào mà đăng ký đều có thể gửi những thông tin độc hại đến ứng dụng của bạn. Có các cách phòng tránh sau đây: Set các quyền là "Nguy hiểm"; set android:exported=false; Local Broadcast

- Chú ý cách đặt tên hành động lắng nghe sao cho tránh trùng lặp tên.
- Không nên chạy task nặng trong hàm `onReceive()` nếu muốn chạy thì chúng ta nên sử dụng một trong hai cách sau: Sử dụng `goAsync()` trong `onReceive()` hoặc sử dụng `JobScheduler`.
- Không nên start Activity từ Broadcast vì lúc này trải nghiệm người dùng đang chập chờn, nhất là khi có nhiều hơn một người nhận. Thay vào đó hãy xem xét hiện thị các thông báo.

2.8. Service

Service là một trong 4 component lớn của Android. Nó là một thành phần hết sức quan trọng, là một Android Developer thì bắt buộc bạn phải nắm rõ và hiểu sâu về Service.

Một Service là một thành phần (component) có thể thực hiện các hoạt động lâu dài trong background và nó không cung cấp một giao diện người dùng. Một thành phần khác của ứng dụng có thể start nó, và nó tiếp tục chạy trong background ngay cả khi người dùng chuyển sang ứng dụng khác. Ngoài ra một thành phần có thể liên kết (bind) với một Service để tương tác với Service đó, thậm chí là thực hiện truyền thông liên tiến trình IPC (interprocess communication - IPC bạn có thể hiểu là một hoạt động chia sẻ dữ liệu qua nhiều tiến trình, thông thường sử dụng giao thức truyền thông và nó phải có Client và Server). Ví dụ: một Service có thể thực hiện các giao dịch mạng, chơi nhạc, ra vào file I/O hoặc tương tác với một content provider, tất cả đều từ background.

2.8.1. Phân loại Service

- Foreground Service.

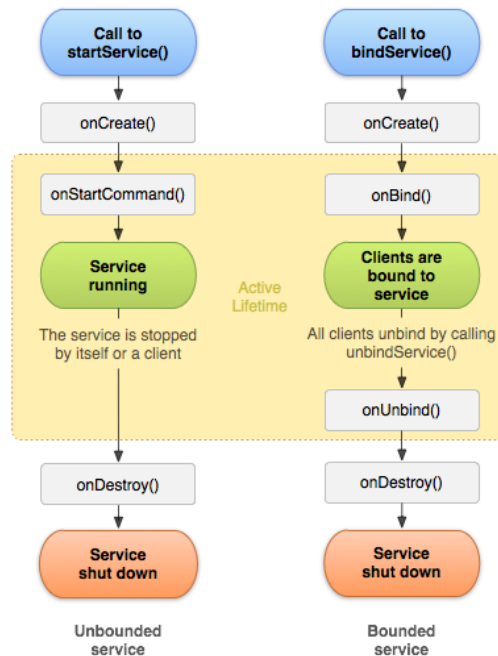
Foreground Service thực hiện một số thao tác mà người dùng chú ý, có thể thấy rõ ràng. Ví dụ một ứng dụng nghe nhạc có thể chơi một bản nhạc và control nó bằng Foreground Service. Một điều bắt buộc Foreground Service phải hiện thị một Notification. Foreground Service sẽ tiếp tục chạy ngay cả khi người dùng không tương tác với ứng dụng.

- Background Service.

Background Service sẽ thực hiện các hoạt động mà không được người dùng chú ý trực tiếp. Ví dụ một ứng dụng sử dụng một service để thu gom bộ nhớ chẳng hạn thì service là một Background Service, hoạt động mà người dùng không cần thiết phải để ý.

- Bound Service.

Service được gọi là Bound khi một thành phần của ứng dụng ràng buộc với nó bởi lời gọi `bindService()`. Một Bound Service cung cấp một giao diện Client - Server cho phép các thành phần tương tác với nó: gửi yêu cầu, nhận kết quả và thậm chí là IPC. Một Bound Service chỉ chạy miễn là có một thành phần ràng buộc với nó. Có thể có nhiều thành phần ràng buộc với Bound Service cùng lúc, nhưng khi tất cả tháo bỏ ràng buộc (unbound) thì nó sẽ Destroy. Trước đây Service thường được chia là Started Service và Bound Service.



Hình 2. 28 vòng đời của service

Một Started Service hay là Unbound Service là service được khởi động bằng phương thức `startService()` từ thành phần khác. Và nó sẽ tiếp tục chạy trong background kể cả khi thành phần khởi tạo nó bị phá hủy. Đây cũng là xem là một Background Service theo cách chia trên.

2.8.2. Độ ưu tiên các loại Service

Hệ thống Android bắt buộc phải dừng một service khi bộ nhớ ít và phải khôi phục tài nguyên hệ thống cho Activity đang được sử dụng. Nếu Service được ràng buộc với một Activity đang sử dụng, nó ít khả năng bị giết; nếu Service được khai báo và chạy ở chế độ Foreground nó cũng khó biết giết. Nếu Service là Started và chạy lâu dài, hệ thống sẽ làm giảm vị trí ưu tiên của nó. Vì phụ thuộc vào process

thì các loại service sẽ được xếp theo độ ưu tiên sau: Bound Service khó bị kill nhất, tiếp theo là Foreground Service và Background Service.

Bound > Foreground > Background

Nếu ở Background Service thì Service dễ bị kill nhất nên ta phải xử lý một cách thích hợp đúng không nào. Tùy thuộc vào giá trị trả về trong onStartCommand() mà Service có thể được khởi động lại.

2.8.3. Các giá trị trả về trong onStartCommand()

Khi Service bị hệ thống kill do thiếu bộ nhớ chẳng hạn, thì dưới đây là 5 giá trị trả về thường dùng trong onStartCommand() để thông báo với hệ thống.

- START_NOT_STICKY

Nếu hệ thống kill service khi giá trị này được trả về thì service này không được khởi động lại trừ khi có một Intent đang được chờ ở onStartCommand(). Đây là lựa chọn an toàn nhất để tránh chạy Service khi không cần thiết và khi ứng dụng có thể khởi động lại một cách đơn giản các công việc chưa hoàn thành.

- START_STICKY

Khi giá trị này được trả về trong onStartCommand, nếu service bị hệ thống kill. Nếu onStartCommand không có một intent nào chờ nó nữa thì Service sẽ được hệ thống khởi động lại với một Intent null.

- START_REDELEVER_INTENT

Nếu Service bị kill thì nó sẽ được khởi động lại với một Intent là Intent cuối cùng mà Service được nhận. Điều này thích hợp với các service đang thực hiện công việc muốn tiếp tục ngay tức thì như download file.

- **START_STICKY_COMPATIBILITY**

Giá trị này cũng giống như START_STICKY nhưng nó không chắc chắn, đảm bảo khởi động lại service.

- **DEFAULT**

Là một sự lựa chọn giữa START_STICKY_COMPATIBILITY hoặc START_STICKY

2.8.4. Các phương thức quan trọng trong vòng đời Service

Khi tạo một service bạn phải kế thừa lớp Service của Android cung cấp. Khi bạn thực thi bạn phải override một vài phương thức quan trọng xử lý trong vòng đời của Service và cung cấp một cơ chế cho phép các thành phần liên kết với Service nếu thích hợp.

- **onStartCommand**

Hệ thống gọi phương thức này khi một thành phần khác (Activity) gọi đến Service bằng câu lệnh startService(). Khi phương thức này được thực hiện, dịch vụ được khởi động và có thể chạy trong background vô thời hạn. Khi công việc hoàn thành bạn nên stop bằng cách gọi stopService() từ một thành phần khác,

hoặc cho chính Service gọi `stopSelf()`. Nếu bạn chỉ muốn ràng buộc buộc với Service thì không nên sử dụng `onStartCommand()`.

- `onBind`

Hệ thống sẽ gọi phương thức này khi một thành phần khác gọi đến Service bằng câu lệnh `bindService()`. Khi bạn triển khai phương thức này bạn phải cung cấp một giao diện để client có thể giao tiếp với Service thông qua một đối tượng `IBinder` do Service trả về. Khi bạn kế thừa từ lớp Service của Android bạn phải luôn luôn override phương thức này, nhưng nếu bạn không muốn ràng buộc (bind) bạn có thể return null.

- `onCreate`

Hệ thống gọi phương thức này khi Service được khởi tạo, và nó chỉ chạy một lần trước khi `onStartCommand()` hoặc `onBind()` được gọi. Nếu Service đã chạy thì phương thức này không được gọi lại lần nào nữa.

- `onDestroy`

Hệ thống gọi phương thức này khi Service không được sử dụng nữa và đang bị hủy (destroy). Bạn cũng nên giải phóng tài nguyên như các `Threads`, `Listeners` hay `Receivers` ở đây. Đây là phương thức cuối cùng được gọi của Service.

2.8.5. *Running một Foreground Service*

Foreground Service phải được thể hiện bởi một Notification với độ ưu tiên `PRIORITYLOW` trở lên, giúp đảm bảo người dùng vẫn biết ứng dụng của mình

đang làm gì ở trong service. Khi Foreground Service đang chạy thì không thể hủy bỏ Notification được. Trừ khi Service bị stopped hoặc được loại bỏ foreground. Để yêu cầu dịch vụ của bạn chạy ở foreground, gọi `startForeground()`. Phương thức này có hai tham số là ID nhận dạng Notification và một Notification.

```
Intent notificationIntent = new Intent(this, ExampleActivity.class);
PendingIntent pendingIntent =
    PendingIntent.getActivity(this, 0, notificationIntent, 0);

Notification notification =
    new Notification.Builder(this, CHANNEL_DEFAULT_IMPORTANCE)
        .setContentTitle(getText(R.string.notification_title))
        .setContentText(getText(R.string.notification_message))
        .setSmallIcon(R.drawable.icon)
        .setContentIntent(pendingIntent)
        .setTicker(getText(R.string.ticker_text))
        .build();

startForeground(ONGOING_NOTIFICATION_ID, notification);
```

Hình 2. 29 Khởi chạy một foreground service

2.8.6. Intent Service

Intent Service là một lớp con của Service, nó là một lớp cung cấp một cấu trúc đơn giản để thực hiện một công việc trên một Thread khác ở background. Điều này cho phép làm một công việc lâu dài mà không ảnh hưởng đến phản hồi của giao diện người dùng. Nó không ảnh hưởng đến các sự kiện từ giao diện người dùng như đối vs AsyncTask chẳng hạn. Sau đây là các hạn chế của một Intent Service:

- Nó không trực tiếp tương tác được với giao diện người dùng. Muốn đưa kết quả lên giao diện ta phải gửi nó đến một Activity.
- Intent Service yêu cầu công việc chạy tuần tự. Nếu có một yêu cầu mới mà Intent Service chưa hoàn thành công việc trước đó thì công việc mới này chưa được bắt đầu mà phải đợi cho công việc trước đó hoàn thành đã.
- Một công việc chạy trong Intent Service không thể bị gián đoạn.

Intent Service là lựa chọn tốt cho các thao tác đơn giản trong background. Để sử dụng một Intent Service bạn phải kế thừa nó.

```
public class RSSPullService extends IntentService {
    @Override
    protected void onHandleIntent(Intent workIntent) {
        // Gets data from the incoming Intent
        String dataString = workIntent.getDataString();
        ...
        // Do work here, based on the contents of dataString
        ...
    }
}
```

Hình 2. 30 tạo một Intent Service

CHƯƠNG 3. MÔ TẢ VÀ PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.1. Tổng quan về Samsung Knox

3.1.1. Samsung Knox

Samsung Knox là một giải pháp bảo mật di động hàng đầu cung cấp một môi trường an toàn cho dữ liệu và ứng dụng của công ty cho tất cả các thiết bị Galaxy. Nó bảo vệ doanh nghiệp và quyền riêng tư cá nhân của bạn khỏi một thiết bị mà không cần bảo vệ CNTT của bên thứ ba.

Vào ngày 5 tháng 3 năm 2018, Samsung đã công bố các thiết bị chạy Knox 3.0 trở lên tích hợp hoàn hảo với các tính năng tương tự của Android Enterprise.

Samsung Knox cung cấp danh sách các tính năng bảo mật cho cả phần cứng và phần mềm. Cho phép các nội dung của cá nhân và doanh nghiệp được bảo mật an toàn trên thiết bị có Knox.

Samsung Knox còn cung cấp các SDK giúp các nhà phát triển tiếp cận dễ dàng hơn.

Knox là nền tảng bảo mật cao cấp của Samsung và là giải pháp an ninh giúp bạn tự do làm việc và vui chơi mọi lúc, mọi nơi. Samsung Knox bao gồm nền tảng được tích hợp ngay trong thiết bị Samsung và đi kèm một bộ giải pháp tận dụng nền tảng này.

3.1.2. Knox SDK

Thông qua SDK Knox, bạn có thể quản lý một bộ tính năng toàn diện trên thiết bị di động Samsung Android. Bạn có thể kiểm soát tài khoản, ứng dụng, kết nối, tính năng tùy chỉnh, cài đặt thiết bị, bảo mật, cài đặt VPN, v.v. Thông qua hàng trăm phương thức API, SDK cung cấp bộ công cụ mạnh mẽ cho các nhà phát triển để tạo ra các giải pháp hấp dẫn và khác biệt. SDK đặc biệt phù hợp với các doanh nghiệp muốn kiểm soát hoàn toàn thiết lập thiết bị của họ và khả năng đáp ứng các yêu cầu khắt khe nhất của công ty và ngành về bảo mật.

Knox SDK mở rộng các chức năng tiêu chuẩn của SDK Android, nhằm cung cấp quyền truy cập chi tiết vào các tính năng của thiết bị, tùy chọn bảo mật, cài đặt, ...

Ví dụ: Một đoạn sử dụng API ngăn quyền mở máy ảnh.

```
EnterpriseDeviceManager edm = EnterpriseDeviceManager.getInstance(context);
RestrictionPolicy restrictionPolicy = edm.getRestrictionPolicy();

try {
    boolean result = restrictionPolicy.setCameraState(false);
    if (true == result) {
        // Camera is disabled and cannot be enabled by user.
    }
} catch (SecurityException e) { Log.w(TAG, "SecurityException: " + e); }
```

3.2. Mô tả và yêu cầu hệ thống

3.2.1. So sánh hệ thống cũ

- **Hệ thống cũ**

Qua việc khảo sát hệ thống cũ tại các cửa hàng việc bảo vệ các mẫu điện thoại trưng bày tại cửa hàng hiện đang sử dụng một thiết bị có tên control box, tất cả các thiết bị được kết nối tới control box qua dây cáp kết nối usb.

Khi thiết bị bị đánh cắp, control box sẽ phát ra cảnh báo cho nhân viên trong cửa hàng.

Việc triển khai lắp đặt control box là hết sức tốn kém và bị cố định lắp đặt ở một chỗ, gây trở ngại cho việc trải nghiệm của người dùng.

Việc dừng hoặc phát video vẫn còn thủ công do nhân viên điều khiển.

Hệ thống bảo mật và hệ thống phát video là hai ứng dụng riêng biệt.

- **Hệ thống mới**

Việc hệ thống mới được tích hợp hai chức năng bảo vệ và phát video vào cùng một hệ thống là cần thiết. Nó giúp nhìn khay ứng dụng ít bị rối và giảm thiểu cài nhiều ứng dụng trên máy.

Hệ thống cảnh báo được tích hợp trên thiết bị, thông báo cho nhân viên khi thiết bị bị đánh cắp.

Việc sản xuất các dây cáp đặc biệt cho hệ thống bảo mật với chi phí rẻ hơn, dễ dàng cho việc di chuyển và cài đặt hơn, không cản trở trải nghiệm của khách hàng.

Việc tự động phát hoặc dừng video sẽ tiết kiệm hiệu năng của thiết bị hơn, tránh lãng phí tài nguyên của thiết bị, gây hao mòn phần cứng.

3.2.2. Mô tả hệ thống

Tích hợp hai ứng dụng bảo vệ điện thoại và phát video giới thiệu, quảng cáo sản phẩm, nhằm tiết kiệm chi phí và giảm thiểu cài nhiều ứng dụng trên máy.

Tiện lợi cho người dùng.

Ứng dụng của em được phát triển cùng team, có bạn Vương Tùng Dương code chính cho phần Knox khóa thiết bị, kích hoạt ứng dụng và hủy kích hoạt ứng dụng. Em xử lý phân đọc video từ bộ nhớ, xử lý thuật toán sắp xếp video, thuật toán đồng bộ video tại một thời điểm, đồng bộ video khi thiết bị ở chế độ không kết nối tới mạng internet, cài đặt thời gian bật tắt video.

Ứng dụng chỉ chạy được khi cắm cáp an ninh do chính samsung sản xuất, khi cắm cáp điện thoại nhận được một broadcast receiver trong đấy có state là trạng thái cắm cáp hay rút cáp, và value đây là giá trị mà được trả về khi cắm đúng cáp an ninh.

Khi điện thoại bị rút khỏi cáp an ninh, một hệ thống cảnh báo được kích hoạt. Sử dụng Samsung Knox để khóa thiết bị. Thiết bị sẽ đổ chuông cảnh báo cảnh báo bằng loa ngoài cho dùng âm lượng đã tắt hoặc đang kết nối với tai nghe, màn hình và các phím sẽ bị khóa cho đến khi cáp được kết nối lại và mở khóa thiết bị.

Khi hệ thống được kích hoạt vì báo động sai hoặc trục trặc, có thể xử lý bằng cách gắn thẻ NFC. Trong trường hợp thiết bị không hỗ trợ mô-đun NFC, cách mới sẽ được áp dụng để hoàn thành chế độ trộm.

Khi ứng dụng được kích hoạt, các chức năng tắt điện thoại, khởi động lại cũng được vô hiệu hóa bằng Knox, nhằm tránh người dùng khôi phục cài đặt gốc bằng download mode.

Hệ thống cần phải đảm bảo được chức năng và giao diện thân thiện với người sử dụng, Xây dựng ứng dụng UI dựa trên One UI của samsung.

Hiệu năng của app chạy đảm bảo không ảnh hưởng đến hệ thống, hỗ trợ các thiết bị có cấu hình thấp.

Video phát khi điện thoại ở chế độ rảnh cần đồng bộ với nhau ở mọi thời điểm. Điện thoại A dừng ở thời điểm i và điện thoại C dừng ở thời điểm j, các video được đồng bộ với nhau ở giây thứ k của video.

Đọc danh sách các video từ bộ nhớ sau khi hoàn tất, sử dụng thuật toán quick sort để sắp xếp lại video theo thứ tự từ a đến z, nhằm tránh trường hợp các video đọc ra một cách không đồng đều và có thể gây sai lệch thời gian hiển thị

Các video có thời lượng ngẫu nhiên và không cần nhất quán, hệ thống cần xử lý với thuật toán tối ưu tìm video bằng thuật toán tìm kiếm nhị phân. Hỗ trợ các trường hợp trong hệ thống có nhiều video và cần đọc ra một cách nhanh nhất để không làm giảm trải nghiệm của người dùng.

Video intro có thể chạy dù thiết bị không kết nối tới mạng internet. Dù không có kết nối internet các video vẫn được chạy đồng bộ với nhau.

Các kiểm thử về bộ nhớ ram khi chạy cũng được kiểm thử, nhằm đánh giá hiệu năng ứng dụng một cách chính xác và khách quan giúp cài đặt vào các thiết bị phù hợp để khách hàng có sự trải nghiệm là hoàn thiện và đồng bộ nhất.

Tính năng thay đổi thẻ NFC cũng được triển khai, tránh người dùng bị hỏng thẻ cũng như nhằm thay thế do thu hồi thẻ về công ty.

Tính năng cài đặt thời gian phát video là một tính năng rất cần thiết và thực dụng. Nó nhằm dừng video chạy khi thiết bị cửa hàng không mở cửa hoặc không hoạt động, tránh việc hao tổn phần cứng khi thiết bị chạy liên tục.

3.2.3. Yêu cầu hệ thống

3.2.3.1. Yêu cầu cấu hình thiết bị

- Nền tảng: các thiết bị của Samsung
- Android: Android 8.0
- Ram: 768MB

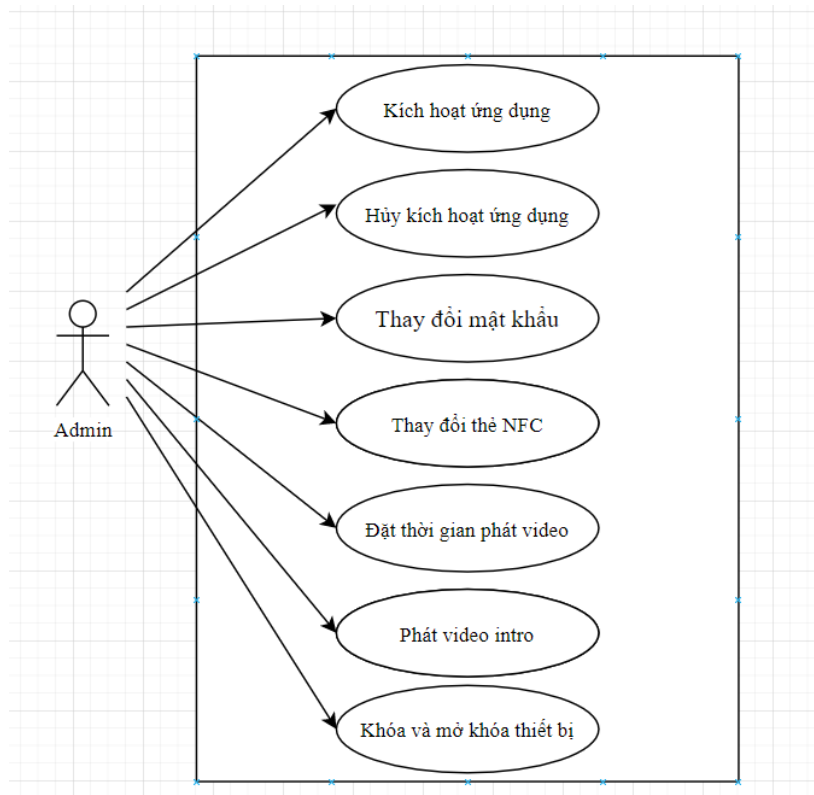
3.2.3.2. Yêu cầu chức năng ứng dụng

- Đăng nhập
- Đăng ký
- Đăng xuất
- Khóa và cảnh báo khi thiết bị bị đánh cắp

- Phát video dưới nền khi điện thoại ở chế độ rảnh
- Quản lý đăng nhập
- Đặt thời gian phát video

3.3. Phân tích thiết kế hệ thống

3.3.1. Biểu đồ use case tổng quát



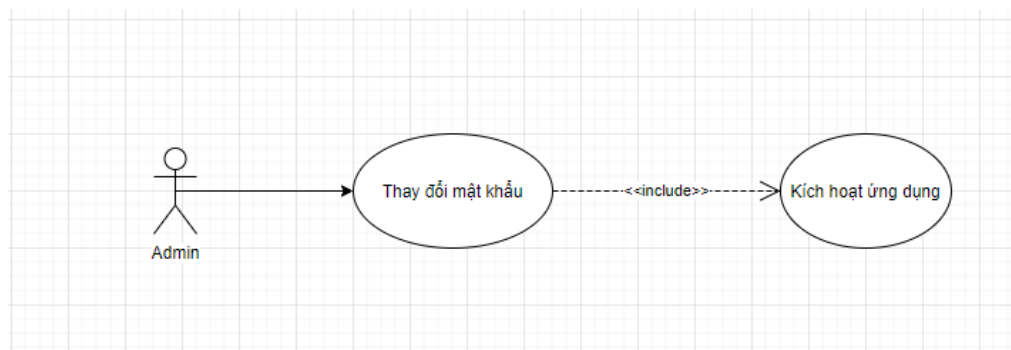
Hình 3. 1 Biểu đồ use case tổng quát

Hình 3.1 cho cái nhìn tổng thể về danh sách các tác nhân và use cases (chức năng) chính của hệ thống. Thông qua hình 3.1, ta có thể thấy hệ thống này chủ yếu phục vụ/đáp ứng các yêu cầu của admin (chủ quản lý cửa hàng/hệ thống bán lẻ) với

7 chức năng chính gồm: Kích hoạt ứng dụng; Hủy kích hoạt ứng dụng; Thay đổi mật khẩu; Thay đổi thẻ NFC; Đặt thời gian phát video; Phát video intro; Khóa và mở thiết bị.

3.3.2. Phân rã use case

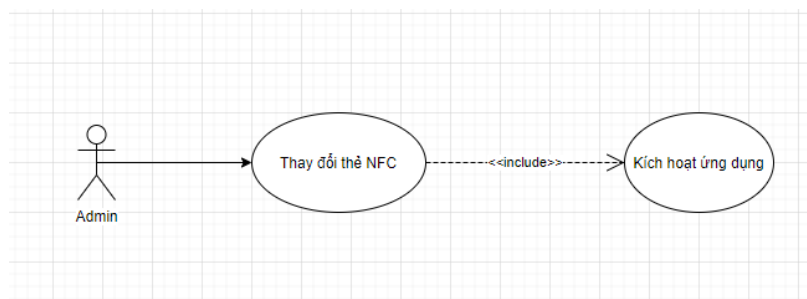
- Phân rã chức năng “Thay đổi mật khẩu”



Hình 3. 2 Biểu đồ phân rã chức năng “Thay đổi mật khẩu”

Hình 3.2 cho cái nhìn tổng thể thay đổi mật khẩu. Chức năng này được sử dụng khi người dùng cần thay đổi mật khẩu ứng.

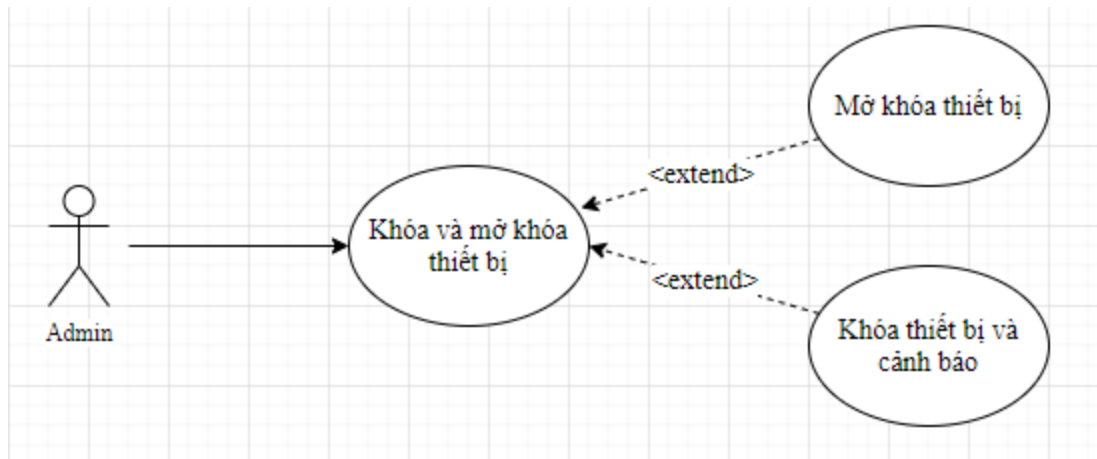
- Phân rã chức năng “Thay đổi thẻ NFC”



Hình 3. 3 Biểu đồ phân rã chức năng “Thay đổi thẻ NFC”

Hình 3.3 cho cái nhìn tổng thể thay đổi thẻ NFC. Chức năng này được sử dụng khi người dùng cần thay đổi thẻ NFC.

- Phân rã chức năng “Khóa và mở khóa thiết bị”

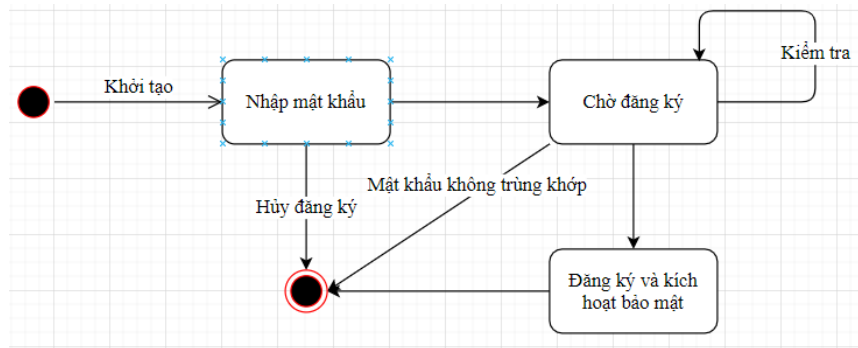


Hình 3. 4 Biểu đồ phân rã chức năng “Khóa và mở khóa thiết bị”

Hình 3.4 cho ta thấy phân rã chức năng khóa thiết bị khi thiết bị bị đánh cắp và chức năng mở khóa thiết bị khi thiết bị đang bị khóa.

3.3.3. Biểu đồ trạng thái

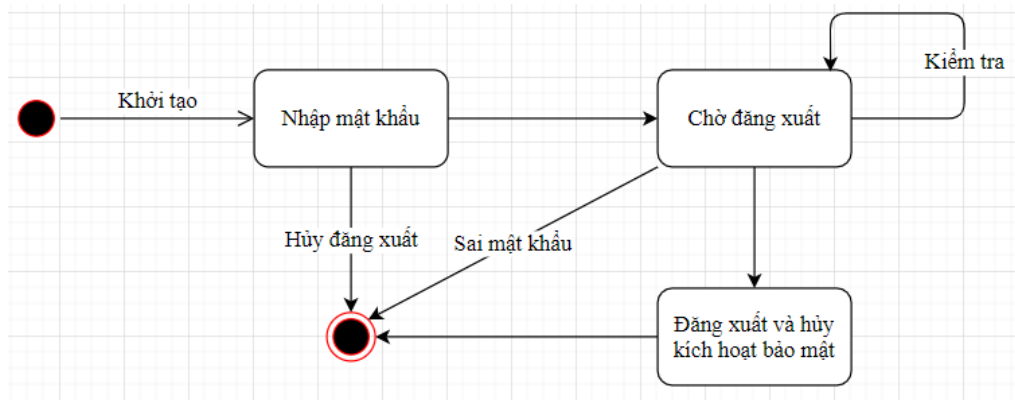
- Biểu đồ trạng thái “Kích hoạt ứng dụng”



Hình 3. 5 Biểu đồ trạng thái “kích hoạt ứng dụng”

Hình 3.5 cho thấy cái nhìn tổng quan quá trình kích hoạt ứng dụng, sau khi ứng dụng được kích hoạt thì chức năng bảo vệ cùng chức năng phát video intro cũng sẽ được kích hoạt.

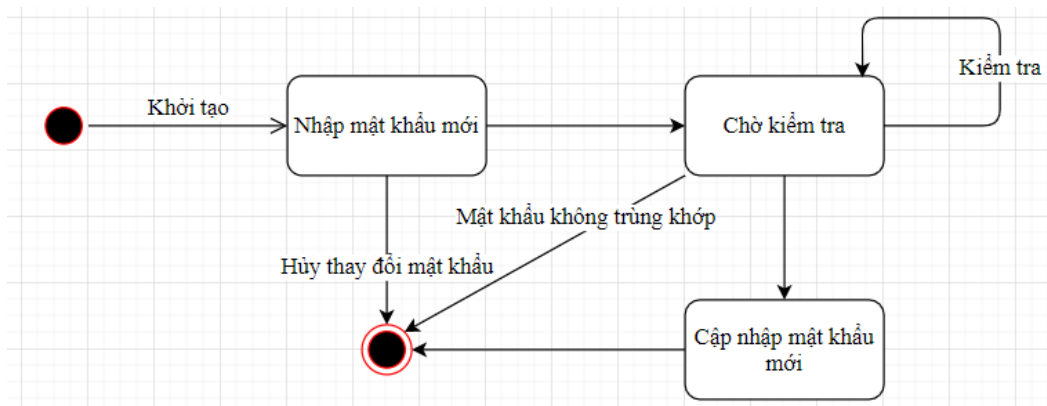
- Biểu đồ trạng thái “Hủy kích hoạt”



Hình 3. 6 Biểu đồ trạng thái “Đăng xuất và hủy kích hoạt”

Hình 3.6 Đây là biểu đồ biểu thị cho việc người dùng muốn gỡ cài đặt ứng dụng hoặc tắt chế độ bảo vệ điện thoại.

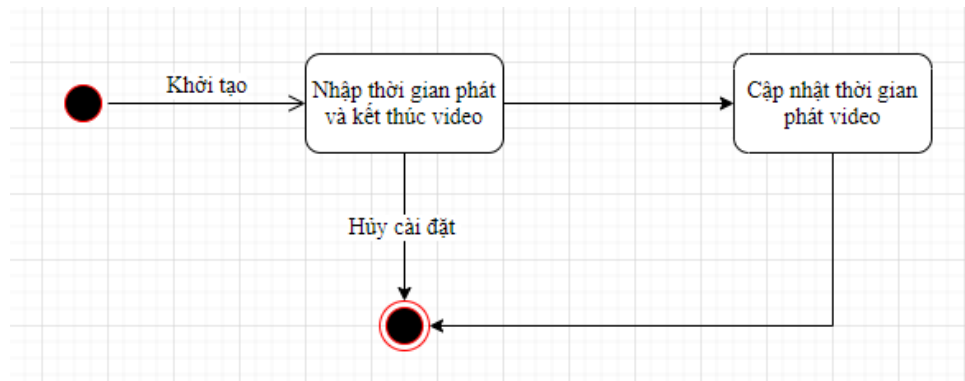
- Biểu đồ trạng thái “Thay đổi mật khẩu”



Hình 3. 7 Biểu đồ trạng thái “Thay đổi mật khẩu”

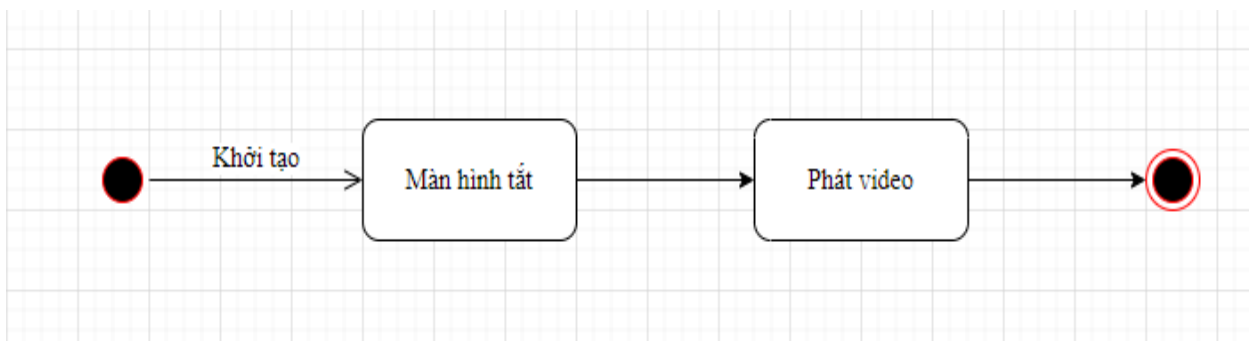
Hình 3.7 Cho thấy cái nhìn tổng quan về việc thay đổi mật khẩu. Mật khẩu ở đây là cách nói chung cho mật khẩu dạng text và mật khẩu dạng NFC.

- Biểu đồ trạng thái “Đặt thời gian phát video”



Hình 3. 8 Biểu đồ trạng thái “Đặt thời gian phát video”

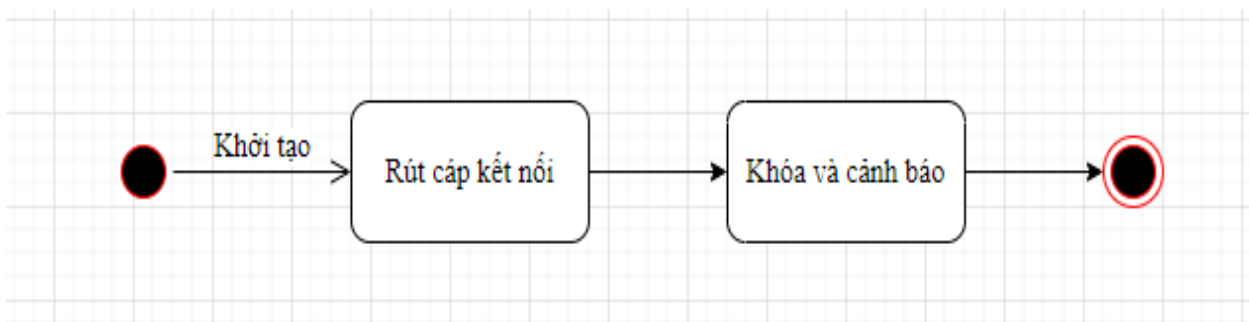
- Biểu đồ trạng thái “Phát video”



Hình 3. 9 Biểu đồ trạng thái “Phát video”

Biểu đồ 3.9 cho thấy cái nhìn tổng quan về việc phát video khi thiết bị ở chế độ rảnh. Khi thiết bị không được sử dụng hoặc người dùng ấn nút tắt màn hình, thì tính năng phát video quảng cáo sẽ được chạy.

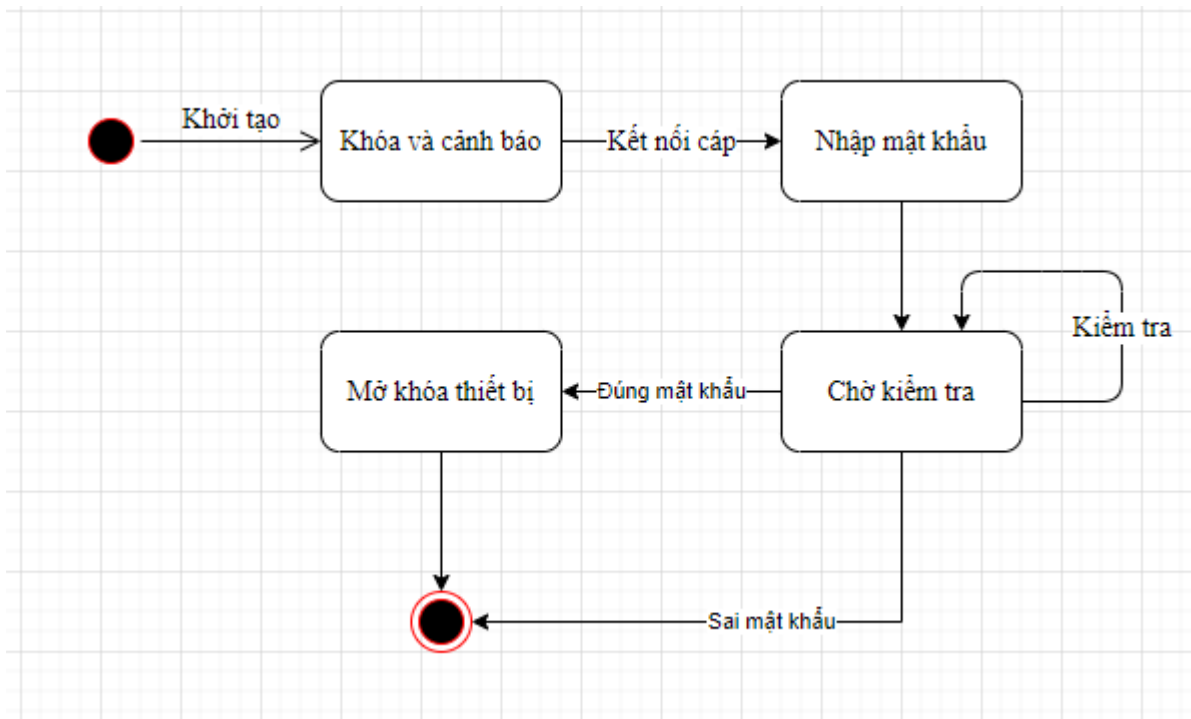
- Biểu đồ trạng thái “Khóa và cảnh báo”



Hình 3. 10 Biểu đồ trạng thái “Khóa và cảnh báo”

Hình 3.10 cho thấy quá trình tổng quan khi thiết bị bị khóa lại và chuông cảnh báo được kích hoạt.

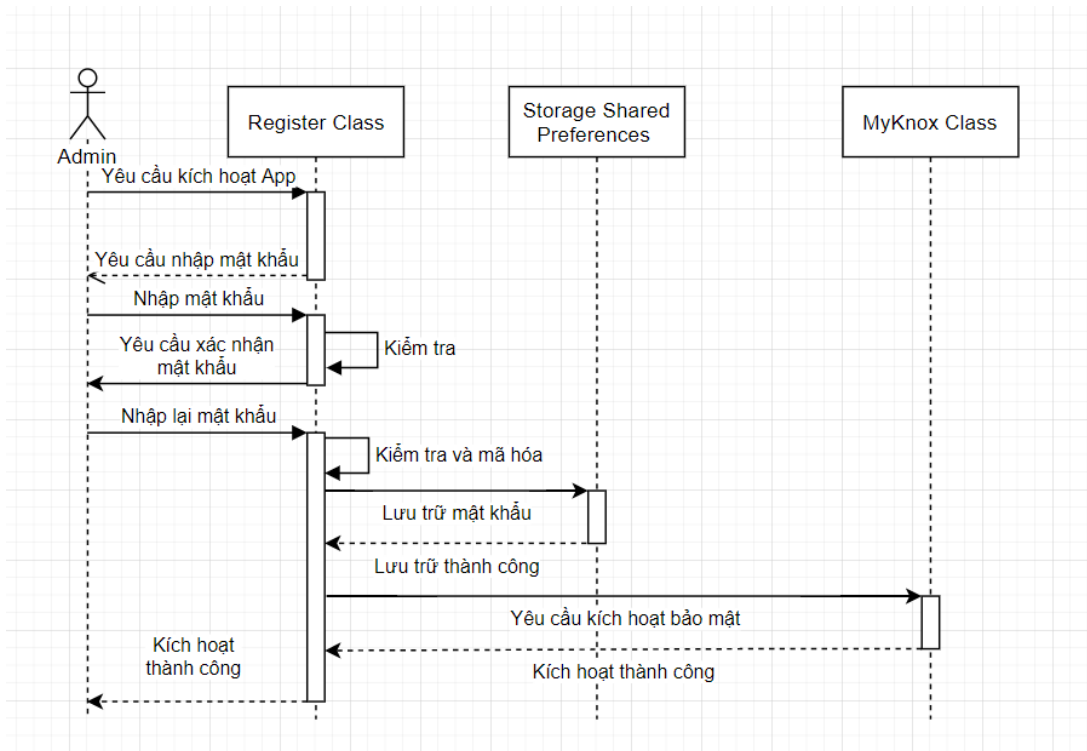
- Biểu đồ trạng thái “Mở khóa thiết bị”



Hình 3. 11 Biểu đồ trạng thái chức năng “Mở khóa thiết bị”

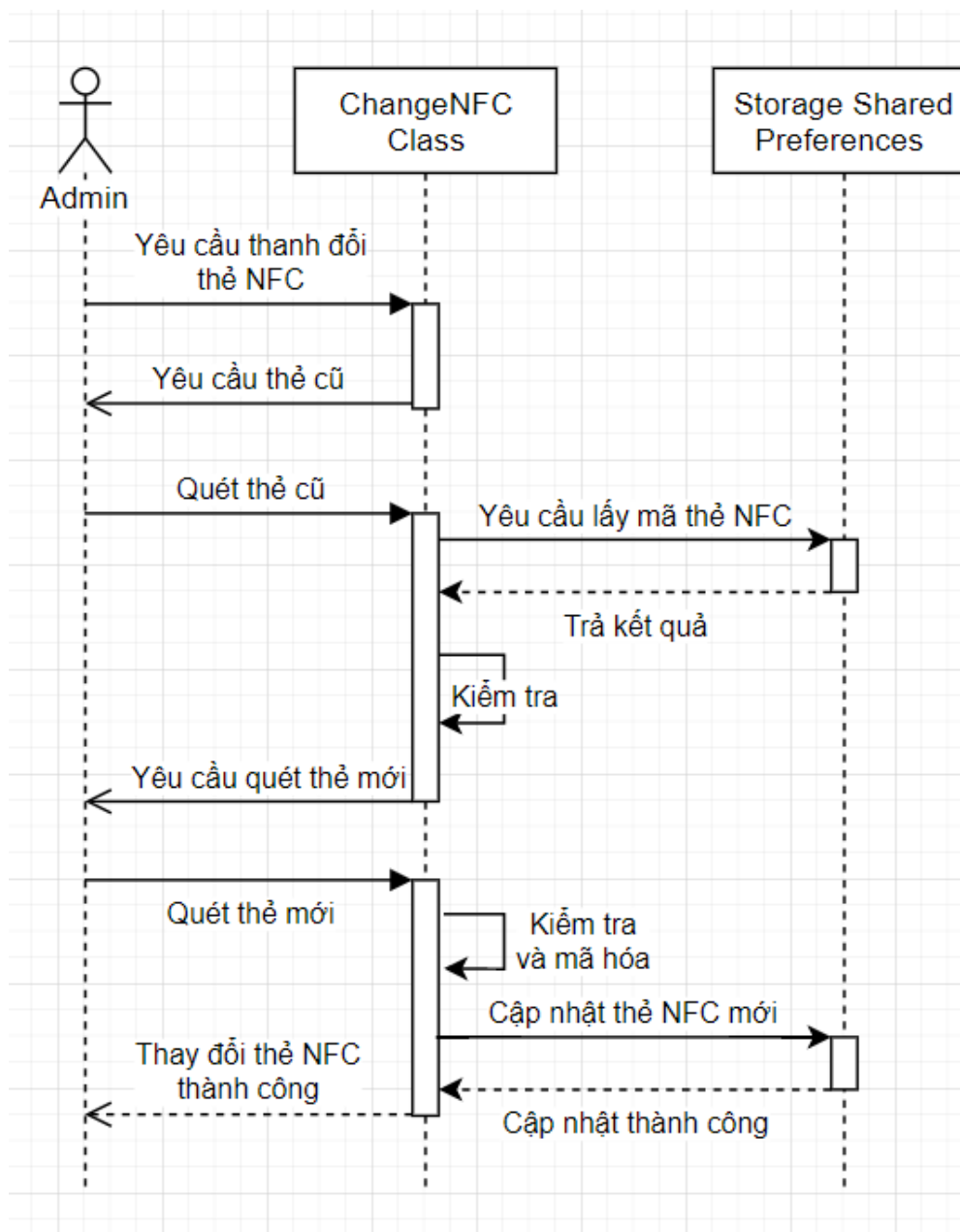
3.3.4. Biểu đồ trình tự

- Biểu đồ trình tự chức năng “Kích hoạt ứng dụng”



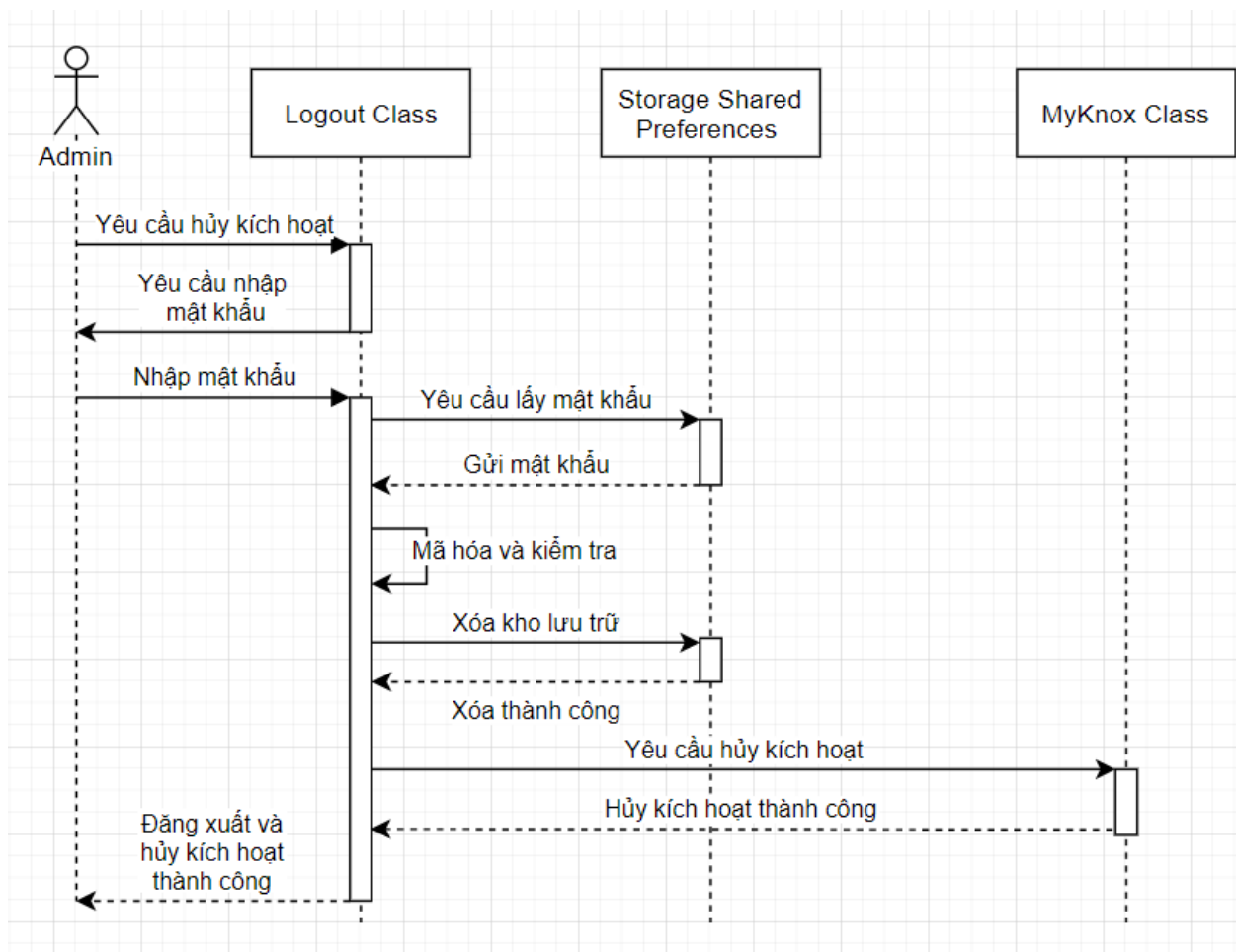
Hình 3. 12 Biểu đồ trình tự chức năng “Kích hoạt ứng dụng”

- Biểu đồ trình tự cho chức năng “Thay đổi thẻ NFC”



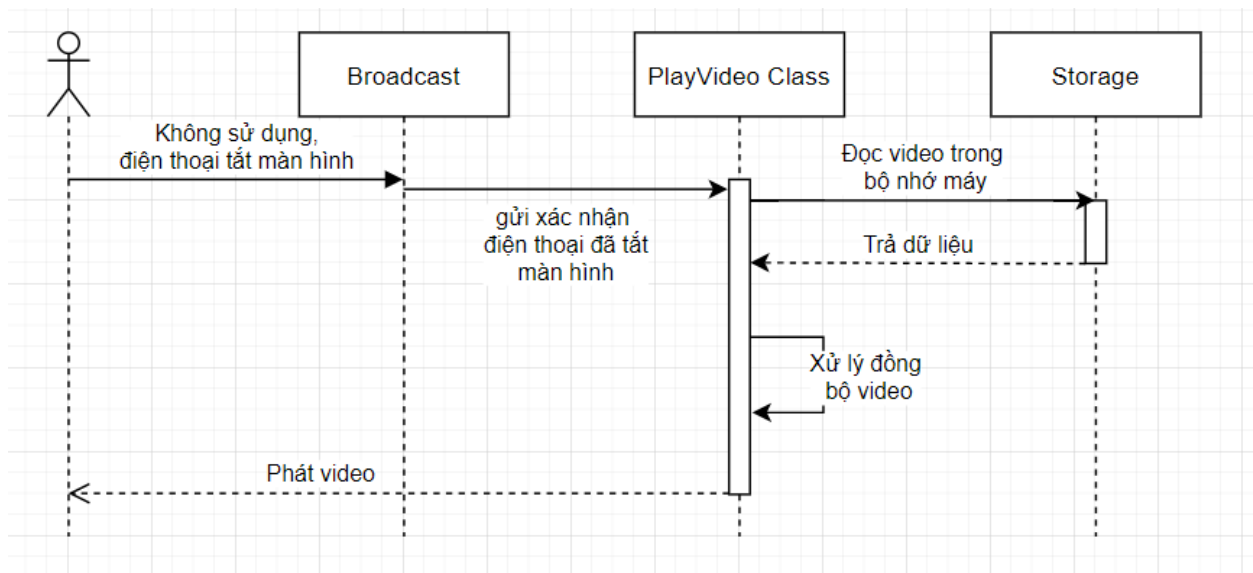
Hình 3. 13 Biểu đồ trình tự cho chức năng “Thay đổi thẻ NFC”

- Biểu đồ trình tự cho chức năng “Hủy kích hoạt”



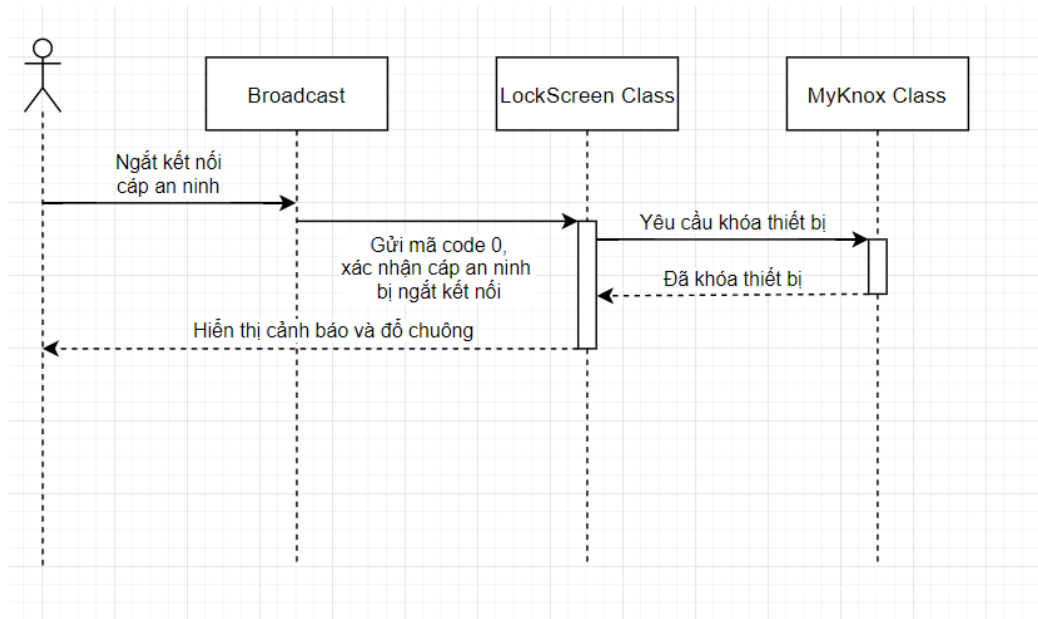
Hình 3. 14 Biểu đồ trình tự cho chức năng “Hủy kích hoạt”

- Biểu đồ trình tự cho chức năng “Phát video intro”



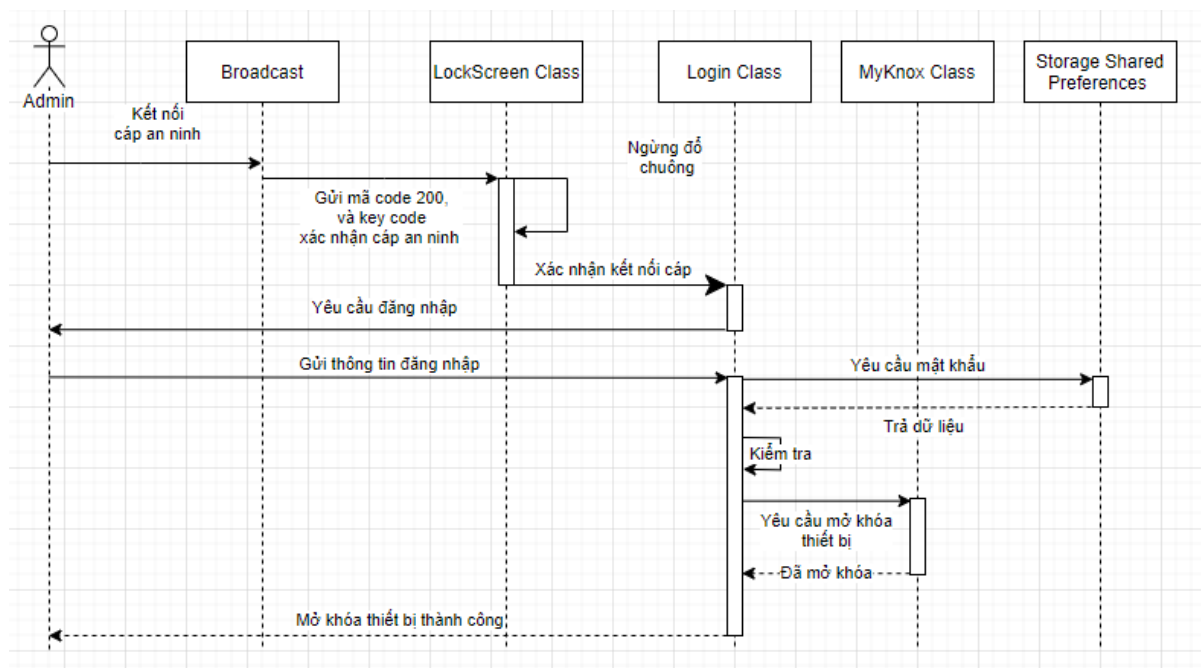
Hình 3. 15 Biểu đồ trình tự cho chức năng “Phát video intro”

- Biểu đồ trình tự cho chức năng “khóa thiết bị”.



Hình 3. 16 Biểu đồ trình tự cho chức năng “khóa thiết bị”.

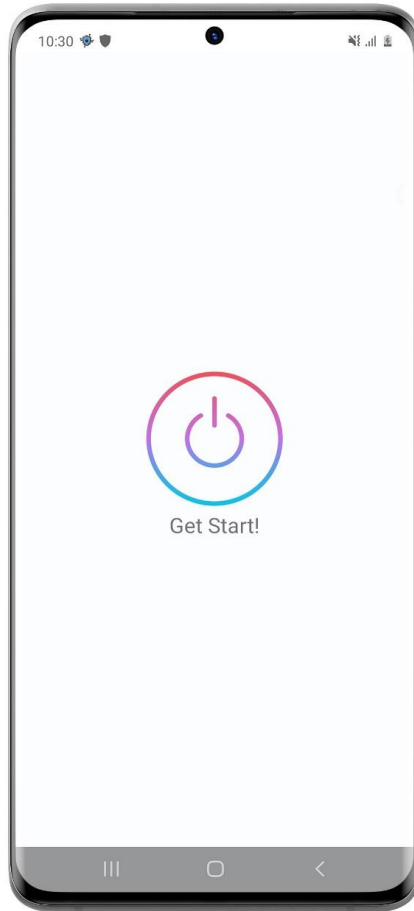
- Biểu đồ trình tự cho chức năng “Mở khóa thiết bị”



Hình 3. 17 Biểu đồ trình tự cho chức năng “Mở khóa thiết bị”.

CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG

4.1. Giao diện bắt đầu khởi chạy ứng dụng



Hình 4. 1 Giao diện bắt đầu khởi chạy ứng dụng

Giao diện khởi chạy là giao diện bắt đầu được chạy ở lần đầu tiên. Trên giao diện có một nút bấm. Khi Admin ấn vào, hệ thống tiếp nhận và bắt đầu khởi chạy hệ

thông. Nếu hệ thống chưa được cấp quyền. Màn hình thông báo xin cấp quyền ứng dụng hiển thị lên.

4.2. Giao diện cấp quyền cho ứng dụng

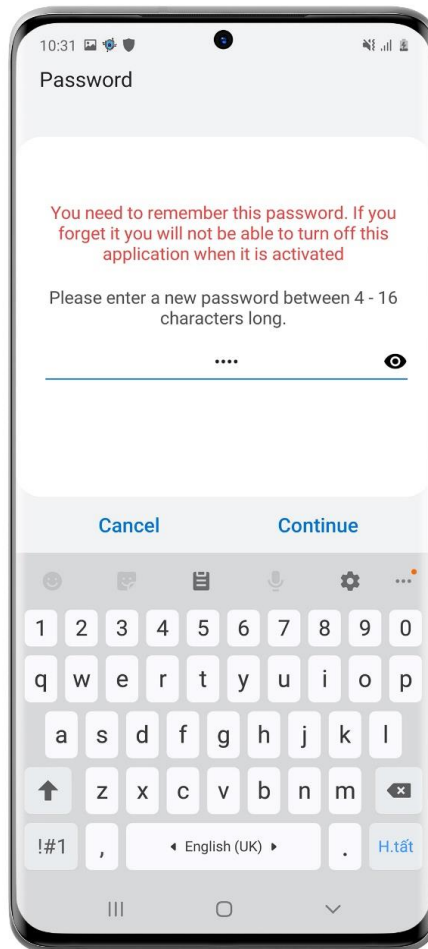


Hình 4. 2 Giao diện xin cấp quyền Admin

Khi ứng dụng chưa được cấp quyền Admin. Giao diện cấp quyền hiện được hiện lên có ghi danh sách các sẽ được cấp nếu người sử dụng dùng ấn vào “Bật”.

Nếu người sử dụng ấn vào “Thoát” thì ứng dụng sẽ kết thúc và sẽ bị buộc dừng. Khi người sử dụng ấn vào gỡ cài đặt. Việc gỡ ứng dụng sẽ được thực hiện.

4.3. Giao diện đăng ký



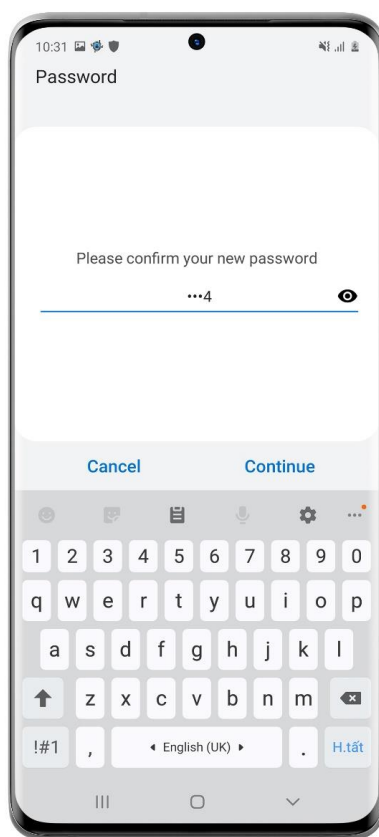
Hình 4. 3 Giao diện đăng ký

Xin các quyền cần thiết xong. Màn hình đăng ký sẽ được hiển thị nếu người sử dụng chưa kích hoạt ứng dụng.

Khi người sử dụng nhập đủ độ dài của mật khẩu, Thì nút “Continue” mới được hiển thị lên, cho phép người sử dụng đến bước tiếp theo.

Ngoài ra còn nút hiển thị mật khẩu và ẩn mật khẩu trong trường hợp khi người dùng muốn xem mật khẩu mình nhập đã chính xác.

4.4. Giao diện xác nhận mật khẩu

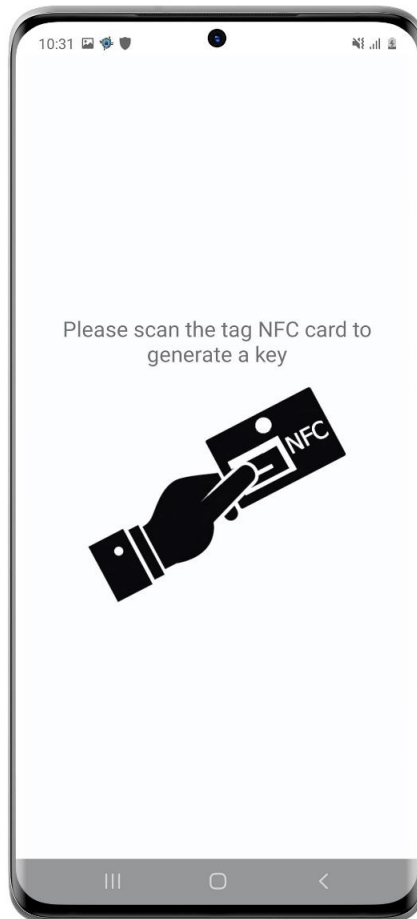


Hình 4. 4 Giao diện xác nhận mật khẩu

Khi người sử dụng nhập mật khẩu xong. Giao diện nhập lại mật khẩu được hiển thị nhằm xác nhận lại mật khẩu người sử dụng vừa nhập, giúp người sử dụng

không nhập sai mật khẩu nhằm tránh quên mật khẩu và nhập mật khẩu không chính xác.

4.5. Giao diện quét thẻ NFC

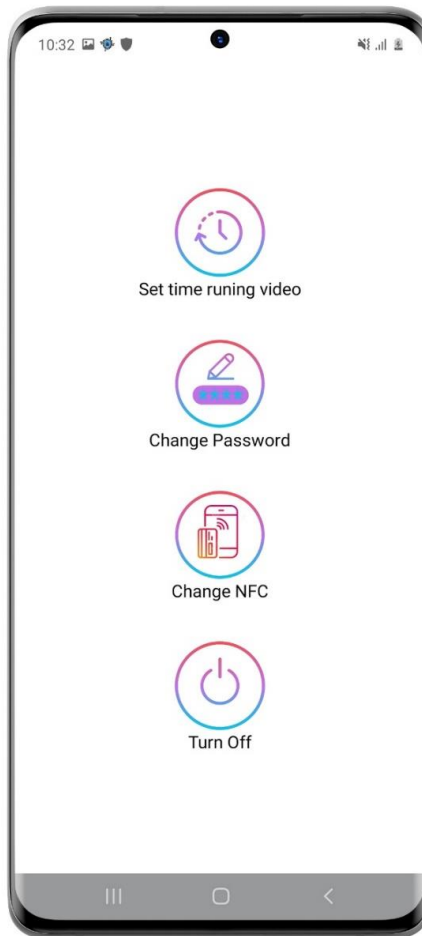


Hình 4. 5 Giao diện đăng ký thẻ NFC

Giao diện đăng ký thẻ NFC cho phép người sử dụng tạo thêm một lớp khóa bảo mật bằng thẻ NFC. Khi người dùng quét thẻ NFC hệ thống sẽ lưu lại.

Nếu người dùng không đăng ký thẻ hoặc thoát ứng dụng. Hệ thống sẽ không lưu lại mật khẩu.

4.6. Giao diện trang chủ

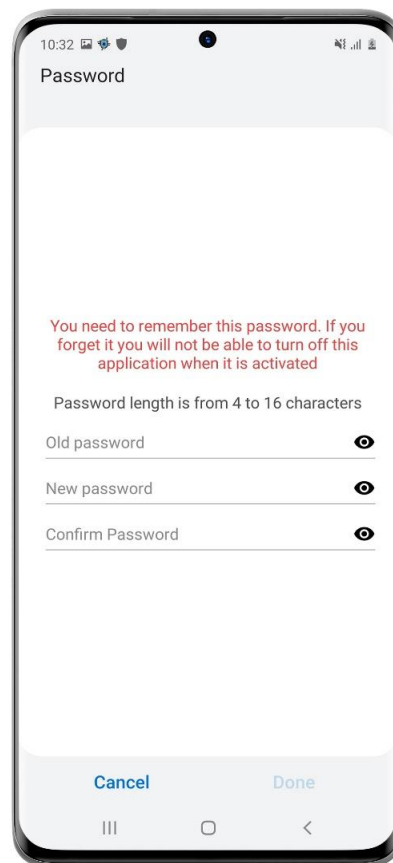


Hình 4. 6 Giao diện trang chủ

Khi việc đăng ký hoặc đăng nhập được hoàn tất. Giao diện trang chủ được hiện thị, ứng dụng sẽ kích hoạt tính năng bảo vệ.

Trên giao diện trang chủ có các nút ấn tương ứng với các chức năng của ứng dụng gồm: Cài đặt thời gian phát video, thay đổi mật khẩu, thay đổi thẻ NFC và hủy kích hoạt ứng dụng.

4.7. Giao diện thay đổi mật khẩu và thẻ NFC



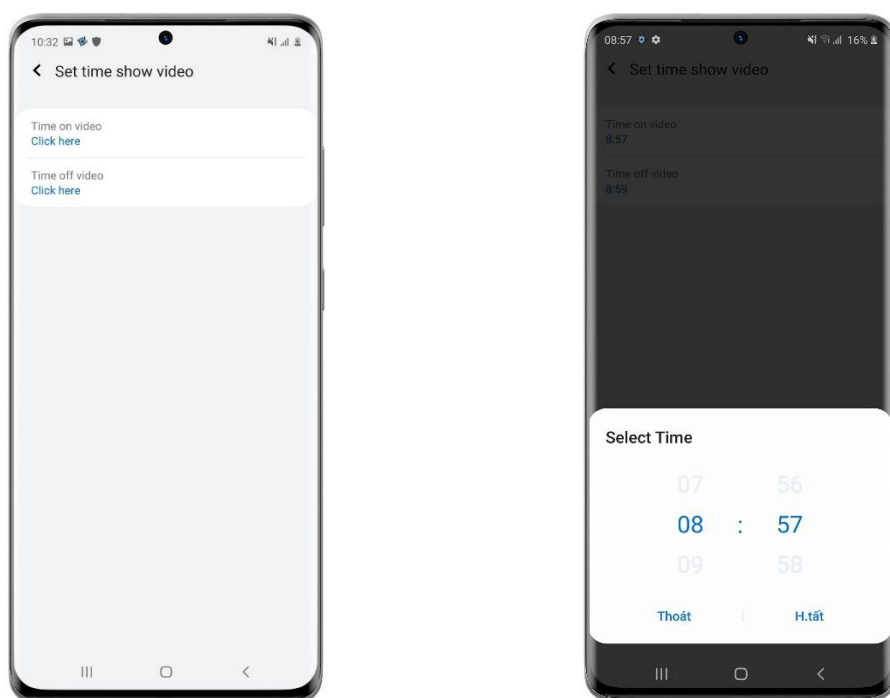
Hình 4. 7 Giao diện thay đổi mật khẩu

Khi độ dài mật khẩu đáp ứng đúng yêu cầu, thì nút “Done” sẽ hiển thị.

Giao diện thay đổi gồm có ba trường nhập dữ liệu. Với trường đầu tiên là mật khẩu cũ. Trường thứ hai là mật khẩu mới và trường thứ ba là nhập lại mật khẩu mới. Nếu các trường nhập mật khẩu hợp lệ thì cập nhập mật khẩu thành công.

Nếu người dùng không muốn thay đổi mật khẩu thì có thể ấn “Cancel” để hủy.

4.8. Giao diện đặt thời gian phát video



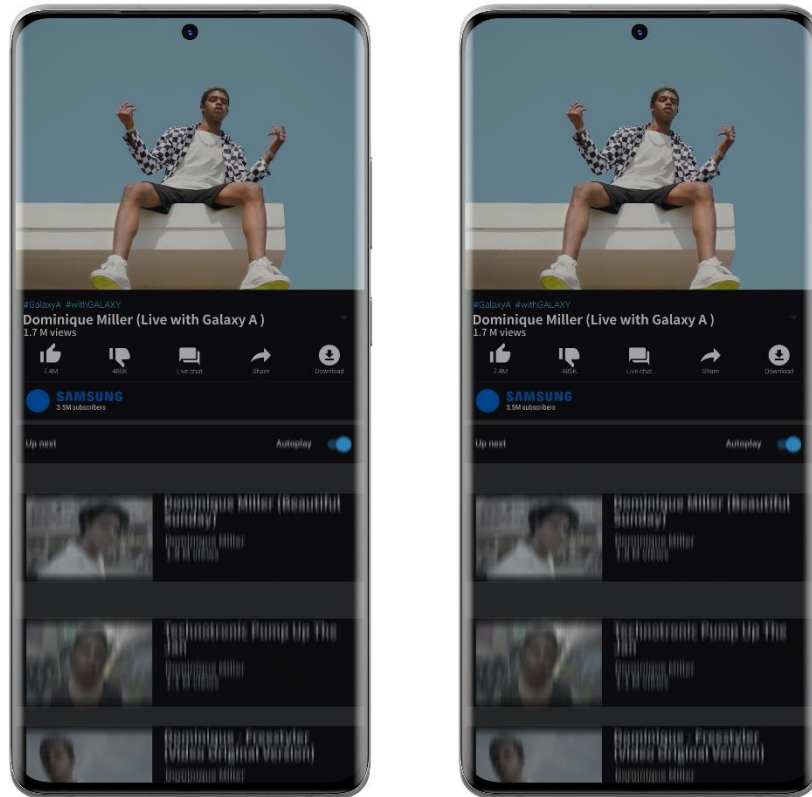
Hình 4. 8 Giao diện đặt thời gian phát video

Người sử dụng muốn đặt thời gian bắt đầu phát video và dừng phát video quảng cáo.

Với giao diện trên, dòng đầu tiên là thời gian bắt đầu phát video. Được tính theo giờ và phút.

Dòng thứ hai là thời gian dừng phát video được tính theo giờ và phút.

4.9. Giao diện phát video



Hình 4. 9 Giao diện phát video

Khi điện thoại tắt màn hình hoặc không sử dụng, một video được chạy quảng cáo sẽ được phát.

Phát video được đồng bộ với các thiết bị cùng loại vào cùng thời điểm.

4.10. Giao diện cảnh báo trộm



Hình 4. 10 Giao diện cảnh báo trộm

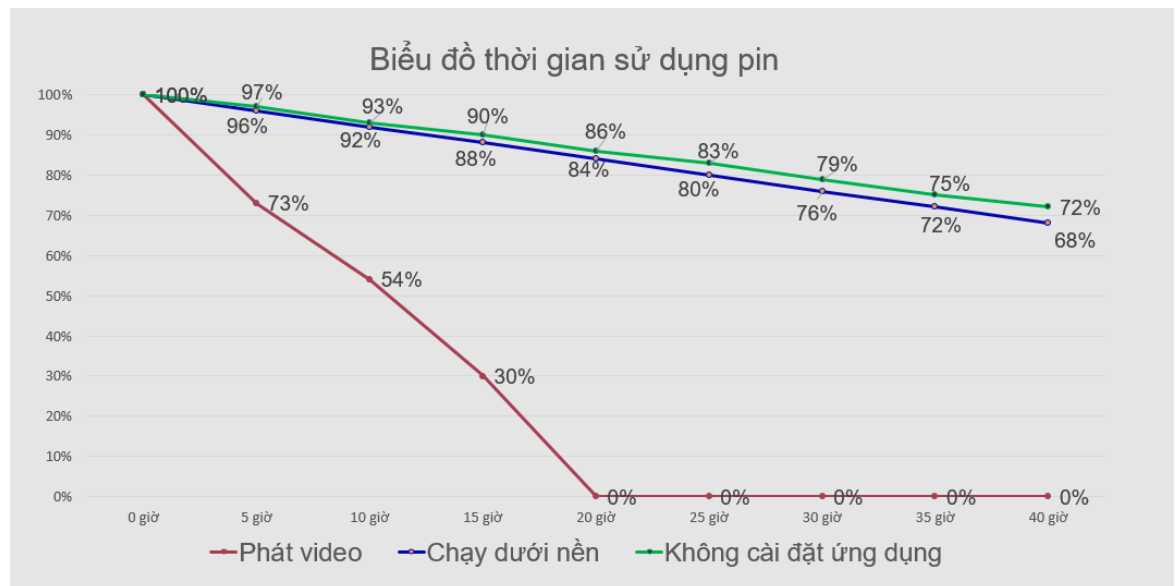
Khi điện thoại bị rút cáp, ứng dụng sẽ được phát chuông cảnh báo ở mức to nhất. Nếu thiết bị đã bị đặt mức chuông mức nhỏ nhất thì nó sẽ được phần mềm cài đặt lại mức âm ở mức to nhất. Ứng dụng chỉ ngừng báo động khi nó được cắm cáp trở lại.

Sau khi người dùng bấm cấp ứng dụng sẽ yêu cầu người dùng làm hai bước để có thể mở khóa máy.

Bước một: Người dùng nhập mật khẩu dạng text mà đã dùng để đăng ký khi thực hiện kích hoạt ứng dụng

Bước hai: người dùng sử dụng thẻ NFC để mở khóa ứng dụng và sử dụng điện thoại trở lại

4.11. Đánh giá thời gian sử dụng pin của ứng dụng



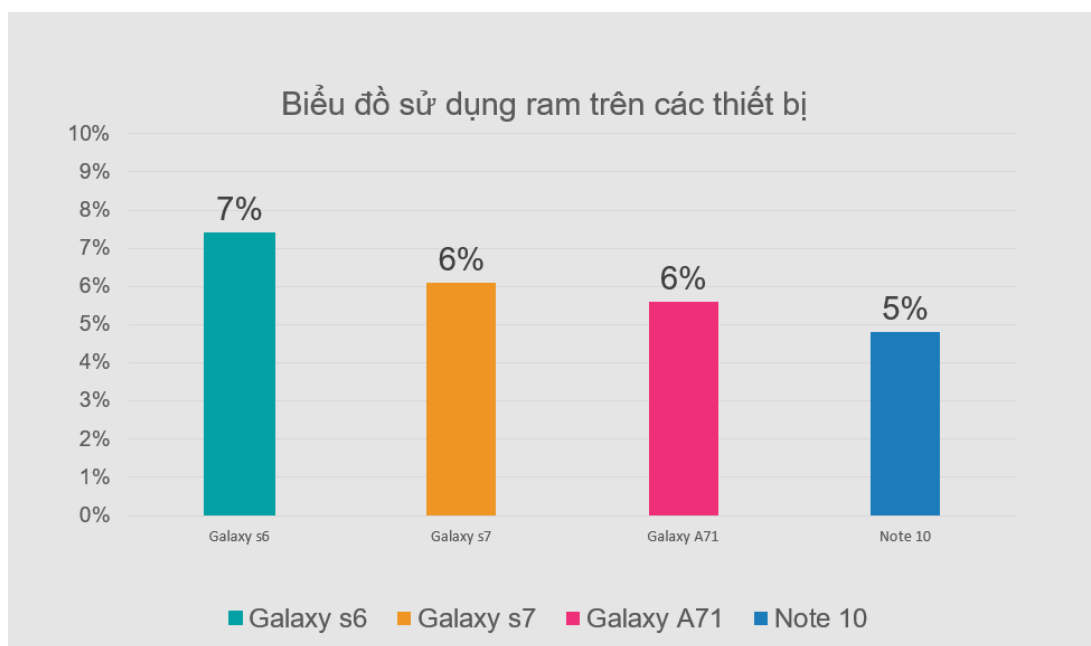
Hình 4. 11 Biểu đồ kiểm thử thời gian sử dụng pin

Quá trình kiểm thử pin thực hiện trên thiết bị samsung galaxy A71. Quá trình kiểm thử sau 5; 10; 15; 20; 25; 30; 35; 40 giờ kiểm thử.

Qua biểu đồ ta thấy được ứng dụng chạy trên thiết bị hoàn toàn đảm bảo không hao tổn phần cứng pin. Đặc biệt hơn khi ta cài ứng dụng ở chế độ không thực hiện phát và đồng bộ video, chỉ thực hiện chức năng bảo vệ điện thoại thì lượng pin tiêu thụ là rất xuất sắc chỉ chênh lệch với khi không cài ứng dụng trung bình là 2.5%.

Thời điểm tốn pin nhất là 4% và ít nhất là 1%. Một mức tiêu thụ hoàn toàn hợp lý và luôn đảm bảo ứng dụng không làm ảnh hưởng đến trải nghiệm của người dùng, và việc hao mòn phần cứng.

4.12. Đánh giá sử dụng ram của các thiết bị khác nhau



Hình 4. 12 Biểu đồ sử dụng ram trên các thiết bị khác nhau

Kiểm thử ram cũng là một vấn đề được bọn em đặt ra. Các thiết bị được kiểm thử ram nó đo chính bằng phần mềm quản lý dung lượng của điện thoại và phần mềm android studio trên máy tính.

Các thiết bị được sử dụng như Samsung Galaxy S6 với 3Gb ram, Samsung Galaxy S7 với 4Gb ram, Samsung Galaxy A71 với 6Gb ram, Samsung Galaxy Note 10 với 8Gb ram.

Các thiết bị dùng để kiểm thử cho thấy ứng dụng hoạt động hoàn toàn ổn định với thiết bị 3Gb chỉ chiếm có 7% và mất 5% với thiết bị 8Gb ram. Trường hợp kiểm thử ở đây em đo khi video intro đang được phát.

KẾT LUẬN

Hệ thống an ninh được cài đặt để ngăn chặn trường hợp trộm cắp và sặc ỏn định cho các thiết bị được trưng bày trong cửa hàng, chạy các video quảng cáo với chi phí rẻ, hệ thống bớt công kênh.

Việc tạo ra một ứng dụng nhằm hỗ trợ việc bảo vệ, quảng cáo sản phẩm là việc hết sức cần thiết. Ứng dụng Samsung Knox được phát triển với mục đích hỗ trợ, bảo vệ và quảng cáo điện thoại đã mang lại những lợi ích lớn cho việc quản lý vào bảo vệ.

Những công việc đã làm được:

Qua quá trình thực hiện đề tài đồ án tốt nghiệp này, em đã thu được những kết quả chính như sau:

- Nắm được kiến thức cơ bản về lập trình Android và các phiên bản Android.
- Xây dựng và phát triển được ứng dụng Samsung Knox với các chức năng chính đáp ứng tốt yêu cầu hỗ trợ việc bảo vệ, quảng cáo sản phẩm/điện thoại cho các cửa hàng/hệ thống bán lẻ của Samsung.
- Tìm hiểu và giải quyết được Android Security.

Một số hạn chế:

- Ứng dụng được tạo ra dành riêng cho các thiết bị samsung. Nên việc cài đặt và chạy trên các thiết bị android khác là không thể.
- Việc sử dụng cấp an ninh do samsung sản xuất là chưa thể thay thế bằng cấp an ninh khác.
- Vấn đề đọc video từ bộ nhớ chưa được thực hiện một cách tự động, cần thực hiện chép video vào thiết bị một cách thủ công.

Hướng phát triển:

- Triển khai Server: Hướng đến triển khai và lưu trữ dữ liệu trên Server, sử dụng Server để điều khiển và đồng bộ giữa các điện thoại. Thực hiện khóa, mở khóa và thay đổi video hoặc đặt thời gian phát video bằng trên Server.
- Thêm các tính năng mới như thông báo tình trạng pin, cảnh báo khi hết pin.
- Thống kê được phiên bản phần mềm của điện thoại ở các cửa hàng.
- Người dùng có thể phản hồi các ý kiến trải nghiệm điện thoại về cho server, để thống kê và cải thiện khi cần thiết.

NHẬN XÉT

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, Ngày Tháng Năm 2020

Giáo viên hướng dẫn

TÀI LIỆU THAM KHẢO

- [1] [https://vi.wikipedia.org/wiki/Android_\(hệ_điều_hành\)](https://vi.wikipedia.org/wiki/Android_(hệ_điều_hành))
- [2] <https://docs.samsungknox.com/dev/knox-sdk/index.htm>
- [3] <https://docs.samsungknox.com/devref/knox-sdk/reference/packages.html>
- [4] <https://developer.android.com/reference>
- [5] <https://stackoverflow.com/>
- [6] <https://www.tutorialspoint.com/index.htm>