Phát triển ứng dụng theo dõi và bảo vệ các mẫu điện thoại thông minh của Samsung tại các cửa hàng bán lẻ.

# LỜI MỞ ĐẦU

Ngày nay với sự phát triển vượt bậc của xã hội cùng với nhu cầu của con người ngày càng cao, ngành CÔNG NGHỆ THÔNG TIN đang dần trở thành nhu cầu không thể thiếu trong xã hội cũng như đời sống con người.

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. Các website quản lý được sử dụng rộng rãi trong sản xuất, học tập, buôn bán… 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 Viêt Nam, các mẫu điện thoại được Samsung sản xuất ngày càng nhiều. Nên việc trưng bày các mẫu điện thoại mới cho người dùng trải nghiệm là rất nhiều, nên việc quản lý và bảo vệ việc tránh đánh cắp điện thoại là một vấn đề đang được đặt ra.

Qua quá trình tìm hiểu, em thấy được các cửa hàng bán lẻ điện thoại như TGDĐ, FPT Shop, Viettel Store, … Hiện tại đều quản lý thiết bị tự động, em cảm thấy tò mò và hấp dễn nên em chọn đề tài này. Ở các hệ thống hiện tại sử các control box, đây là bộ hub kết nối tới thiết bị di động qua cap usb, các hub như này rất tốn kém và cồng kềnh. Thay vì đó bọn em phát triển ứng dụng chỉ sử dụng cap usb có gắn chip ở đầu để tiết kiệm chi phí, kèm theo tích hợp tính năng phát video quảng cáo khi điện ở chế độ rảnh.

Nên em mới thực hiện việc *“Phát triển ứng dụng theo dõi và bảo vệ các mẫu điện thoại thông minh của Samsung tại các cửa hàng bán lẻ”*. Nhằm bảo vệ và theo dõ các các mẫu điện thoại của Samsung tại các cửa hàng trưng bày và bán lẻ.

Trong quá trình thực hiện phát triển em vẫn còn thiếu kinh nghiệm, vì vậy không thể tránh khỏi khỏi những sai sót, em mong nhận được sự đóng góp ý kiến, phê bình của thầy cô.

# 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 tỉ 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.

## **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 điêu 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.

### Android khác với các hệ điêu hành chạy trên thiết bị di động khác

Android đã thu hút được sự chú ý của giới công nghệ khác toàn cầu khi đứa 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.

### 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 Phone7 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 cức 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.

## **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 thế không cần hiểu rõ cấu trúc của một hệ điều hành những chúng ta vẫn có thế 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.

### Android Plastform

Bao gồm hệ điều hành Android đầu 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 Frameword
* Android Runtime
* Native Librarites
* Linux Kernel

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

### 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

### Tầng Native Libraries

* System C library: Có nguồn gốc từ hệ thông thư việc chuẩn C (libc), điều chỉnh các thiết bị nhúng trên Linux.
* Media Libraries: Mở rộng từ PacketVideo’s OpenCORE. Là thử viện hộ trợ playback và recording của nhiều định dạng video và image phổ biến như MPEG4, H.264, MP3, AAC, JPG và PNG...
* Surface Manger: 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
* SQL lite: Quản lý database của ứng dụng

### Tầng Runtime

Một ứng dụng Android chạy trên một proccess 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 dang **\*\*.dex** (Dalvik Excutable), 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 dược compile trước đó bởi 1 trinh biên dịch ngôn ngữ Java), sở dĩ máy ảo chạy được các class này là nhờ chương trình Dx tool đã convert các class sang định dang **.dex**.

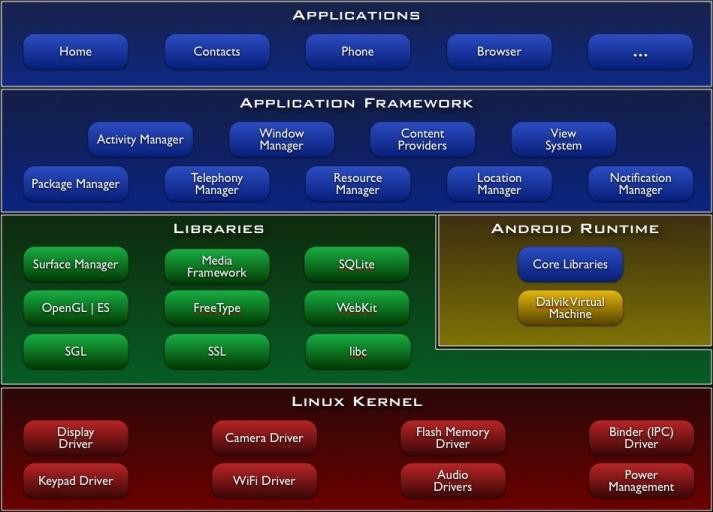
### Tầng ApplicationFramework

Đây là tầng mà Google xây dựng cho phép các Developer để phát triển các ứng dụng của họ trnee Android, chỉ bằng cách gọi cách API có sẵn mà Google đã viết để sử dụng các tính năng của phần cứng mà không cân 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 cứ, 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 dịnh vụ và hệ thống cơ bản sau:

* View UI: dùng để xây dựng layout của ứng dụng bao gồm: list view, 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. Kiến trúc Android

# CHƯƠNG 2. MÔI TRƯỜNG VÀ CÔNG CỤ LẬP TRÌNH

## **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 mền Intelli 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.

## **Cài đặt Android Studio**

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



Hình 2. Trang chủ tải Android Studio

Bước 2: Tiến hành cài đặt trên Windows. Sau khi tải xuống sau tai 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. 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. Chọn Install để cài đặt Android Studio

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



Hình 2. Hoàn thành việc cài đặt Android Studio



Hình 2. Đã hoàn thành việc cài đặt Android Studio

## **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. Bắt đầu tạo project mới

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



Hình 2. 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 trự 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 song chúng ta chọn Next để tiếp tục.



Hình 2. 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. Chọn Manager AVD

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



Hình 2. Create Virtual Device

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



Hình 2. Chọn máy ảo

Bước 7: Chọn hệ điều hành cho máy ảo.



Hình 2. Chọn hệ điều hanh cho máy ảo.

Bước 8: Chọn vào http://www.hiepsiit.com/public/uploads/images/android/introduction/bieutuong.png để bắt đầu khởi động máy ảo.



Hình 2. Danh sách máy ảo



Hình 2. Máy ảo đã khởi động xong

Bước 9: Giờ ta có thể tiến hành Build app bằng cách chọn vào biếu tượng http://www.hiepsiit.com/public/uploads/images/android/introduction/bieutuong.png để bắt đầu Build.



Hình 2. Toolbar Build

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



Hình 2. Chương trình đầu tiên

## **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 dư liệu giữa các ứng dụng của Android
* Intent: Là nền tảng để truyền tải các thông báo. Intent được sử dụng để gửi cac thông báo đi nhằm khởi tạo 1 Activity hay Service để thực hiện công việc mà chúng mong muốn.

## **Chu kỳ số của ứng dụng Android**

### Chu kỳ sống

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

### 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 đinh 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. Activity Stack

### Các trạng thái của chu kỳ sống



Hình 2. 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 bảo 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 Paused cũng có thể bị giải phóng.

### 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(): đưọc gọi khi ứng dụng chuyển sang onStop(), nhưng muốn khởi động lại bằng onStart().

## **Các Layout trong Android**

### 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 tai 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. Minh họa FrameLayout

### 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:android1="http://schemas.android.com/apk/res/android"  
 android1:layout\_width="match\_parent"  
 android1:layout\_height="match\_parent"  
 android1:orientation="horizontal"** >  
  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="1"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:textStyle="bold"  
 android1:background="@color/colorAccent"** />  
  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="2"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:textStyle="bold"  
 android1:background="@color/colorPrimary"** />  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="3"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:textStyle="bold"  
 android1:background="#8c0520"** />  
</**LinearLayout**>



Hình 2. Minh họa LinearLayout

### Realative 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 vớ 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. 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**>

### 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**>  
 <**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. Minh họa TableLayout

### GridLayout

GirdLayout 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"  
 android1:layout\_width="match\_parent"  
 android1:layout\_height="match\_parent"  
 android1:columnCount="2"  
 android1:rowCount="2"**>  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="1"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:background="@color/colorAccent"** />  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="2"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:background="@color/colorPrimary"** />  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:text="3"  
 android1:textColor="#fff"  
 android1:textSize="15pt"  
 android1:textAlignment="center"  
 android1:background="#8c0520"** />  
 <**TextView  
 android1:layout\_width="80dp"  
 android1:layout\_height="80dp"  
 android1:background="#efcd21"  
 android1:text="4"  
 android1:textAlignment="center"  
 android1:textColor="#fff"  
 android1:textSize="15pt"** />  
</**GridLayout**>



Hình 2. Minh họa GridLayout

GirdLayout được quy định bằng hai thuộc tính

**android1:columnCount="2" // số hàng  
android1:rowCount="2" // số cột**

# CHƯƠNG 3. GIỚI THIỆU VỀ SAMSUNG KNOX

## **Tổng quan về Samsung Knox**

Samsung Knox là một bộ tính năng bảo mật toàn diện cho cá nhân và doanh nghiệp được cài sẵn trong hầu hết các thiết bị di động thông minh, mính tính bảng và thiết bị đeo của Samsung.

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.

Samsung Knox cung cấp danh sách các tinh 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 SKD giúp các nhà phát triển tiếp cận dễ dàng hơn.

## **Knox SDK**

Samsung Knox SDK cung cấp hàng trăm API để trao quyền bạn phát triển các ứng dụng đầy đủ tính năng cho các thiết bị Samsung.

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); }

# CHƯƠNG 4. MÔ TẢ VÀ PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## **Mô tả và yêu cầu hệ thống**

### Mô tả hệ thống

Với việc nhiều điện thoại của Samsung được trưng bày ở các của hàng bán lẻ. Việc bảo vệ và theo dõ các mẫu điện thoại của Samsung là hết sức cần thiết.

Việc các mẫu điện thoại trưng bày ở của hàng bán lẻ cần được bảo vệ và có thể tự phát video khi người dùng không sử dụng điện thoại là một việc cần thiết.

Khi điện thoại bị rút khỏi cáp kết nối, một hệ thống cảnh báo được kích hoạt.

### Yêu cầu hệ thống

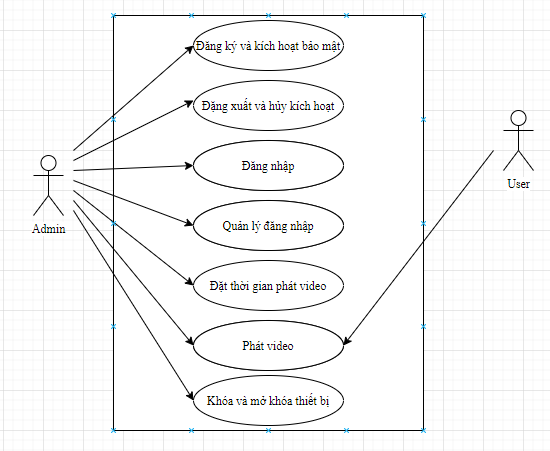
Hệ thống cần phải đáp ứng được tính bảo vệ các mẫu điện thoại của Samsung. Việc phát video và dừng video khi của hàng đóng cửa.

Các chức năng của ứng dụng:

* Đăng nhập
* Quản lý đăng nhập
* Hệ thống khóa và cảnh báo
* Hệ thống tự động phát video
* Quản lý phát video

## **Phân tích thiết kế hệ thống**

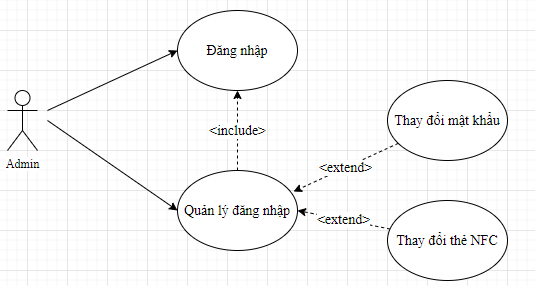
### Biểu đồ use case tổng quát



Hình 4. 1 Biều đồ use case tổng quát

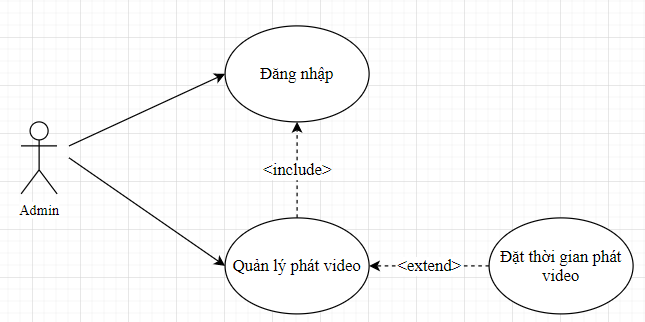
### Phân rã use case

* Phân rã chức năng “Quản lý đăng nhập”



Hình 4. 2 Biểu đồ phân rã chức năng đăng nhập

* Phân rã chức năng “Quản lý phát video”



Hình 4. 3 Biều đồ phân rã chức năng quản lý phát video

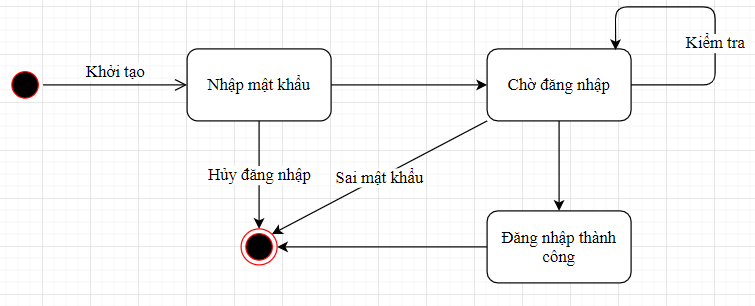
* Phân rã chức năng “Khóa và mở khóa thiết bị”



Hình 4. Biểu đồ phân rã chức năng khóa và mở khóa thiết bị

### Biểu đồ trạng thái

* Biểu đồ trạng thái “Đăng nhập”



Hình 4. 5 Biểu đồ trạng thái “Đăng nhập”

* Biểu đồ trạng thái “Đăng ký và kích hoạt bảo mật”



Hình 4. 6 Biểu đồ trạng thái “Đặng ký và kích hoạt bảo mật”

* Biểu đồ trạng thái “Đăng xuất và hủy kích hoạt”



Hình 4. 7 Biểu đồ trạng thái “Đăng xuất và hủy kích hoạt”

* Biểu đồ trạng thái “Thay đổi mật khẩu”



Hình 4. 8 Biểu đồ trạng thái “Thay đổi mật khẩu”

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



Hình 4. 9 Biểu đồ trạng thái “Đặt thời gian phát video”

* Biểu đồ trạng thái “Phát video”



Hình 4. 10 Biểu đồ trạng thái “Phát video”

* Biểu đồ trạng thái “Khóa và cảnh báo”



Hình 4. Biểu đồ trạng thái “Khóa và cảnh báo”

* Biểu đồ trạng thái “Mở khóa thiết bị”



Hình 4. Biểu đồ “Mở khóa thiết bị”

# CHƯƠNG 5. XÂY DỰNG ỨNG DỤNG