BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGÀNH CÔNG NGHỆ THÔNG TIN

Sinh viên :Vũ Ngọc Đông

Giảng viên hướng dẫn: Ths. Nguyễn Thị Xuân Hương

HẢI PHÒNG – 2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

XÂY DỰNG ỨNG DỤNG ĐĂNG KÝ ĂN TRƯA TẠI TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH CÔNG NGHỆ THÔNG TIN

Sinh viên : Vũ Ngọc Đông

Giảng viên hướng dẫn:Ths. Nguyễn Thị Xuân Hương

HẢI PHÒNG – 2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: **Vũ Ngọc Đông** **Mã SV**: 1612112004

Lớp : CT2001C

Ngành :Công nghệ Thông tin

Tên đề tài: **Xây dựng ứng dụng đăng ký ăn trưa tại trường Đại học Quản lý và Công nghệ Hải Phòng**

NHIỆM VỤ ĐỀ TÀI

1. Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp

- Tìm hiểu về hệ điều hành Android,
- Tìm hiểu về môi trường lập trình Android.
- Xây dựng ứng dụng Đăng ký ăn trưa tại trường Đại học Quản lý và Công nghệ Hải Phòng, trong đó cho phép
    - Thực hiện phân quyền cho người dùng
    - Người dùng đăng nhập hệ thống để đăng ký ăn trưa
    - Thống kê và hiển thị danh sách giảng viên sinh viên đăng ký ăn trưa theo ngày

2. Các tài liệu, số liệu cần thiết

- Số liệu: theo thông tin giảng viên của trường Đại học Quản lý và Công nghệ Hải Phòng

3. Địa điểm thực tập tốt nghiệp

- Trường Đại Quản lý và Công nghệ Hải Phòng

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN TỐT NGHIỆP

Họ và tên : Nguyễn Thị Xuân Hương

Học hàm, học vị : Thạc sỹ

Cơ quan công tác : Trường Đại học Quản lý và Công nghệ Hải Phòng

Nội dung hướng dẫn:

- **Tìm hiểu về hệ điều hành Android**
- **Tìm hiểu về môi trường lập trình Android**
- **Xây dựng ứng dụng đăng ký ăn trưa tại trường Đại học Quản lý và Công nghệ Hải Phòng**

Đề tài tốt nghiệp được giao ngày 30 tháng 03 năm 2020

Yêu cầu phải hoàn thành xong trước ngày 30 tháng 06 năm 2020

Đã nhận nhiệm vụ ĐTTN

Sinh viên

Đã giao nhiệm vụ ĐTTN

Giảng viên hướng dẫn

Hải Phòng, ngày tháng năm 2020

HIỆU TRƯỞNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN TỐT NGHIỆP

Họ và tên giảng viên: ……...

Đơn vị công tác: ……..

Họ và tên sinh viên: …… Ngành: ……

Nội dung hướng dẫn: ……

……..

1. Tinh thần thái độ của sinh viên trong quá trình làm đề tài tốt nghiệp

……....

……....

……....

……....

2. Đánh giá chất lượng của đồ án/khóa luận (so với nội dung yêu cầu đã đề ra trong nhiệm vụ Đ.T. T.N trên các mặt lý luận, thực tiễn, tính toán số liệu…)

……....

……....

……....

……....

3. Ý kiến của giảng viên hướng dẫn tốt nghiệp

Đạt ☐ Không đạt ☐ Điểm:……...

Hải Phòng, ngày 28 tháng 06 năm 2020

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên

QC20-B18

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN CHẤM PHẢN BIỆN

Họ và tên giảng viên: ……

Đơn vị công tác: ……

Họ và tên sinh viên: …… Ngành: ……

Đề tài tốt nghiệp: ……..

…….....

1. Phần nhận xét của giảng viên chấm phản biện

.....

.....

.....

.....

2. Những mặt còn hạn chế

.....

.....

.....

.....

3. Ý kiến của giảng viên chấm phản biện

Được bảo vệ ☐ Không được bảo vệ ☐ Điểm:……

Hải Phòng, ngày …… tháng ..... năm 2020

Giảng viên chấm phản biện

(Ký và ghi rõ họ tên)

QC20-B19

MỤC LỤC

LỜI MỞ ĐẦU

Hiện nay Công nghệ thông tin là một ngành có nhiều đóng góp to lớn cho việc thực hiện các nhiệm vụ mọi lĩnh vực trong cuộc sống.. Các chương trình được tạo ra không chỉ có thể thực hiện được trên máy tính mà còn có thể áp dụng cho các thiết bị vô cùng nhỏ gọn và tiện dụng như điện thoại di động hay máy tính bảng. Do đó việc xây dựng các ứng dụng cho điện thoại di động đang là một ngành công nghiệp mới đầy tiềm năng và hứa hẹn nhiều sự phát triển vượt bậc của ngành khoa học kỹ thuật.

Phần mềm, ứng dụng cho điện thoại di động hiện nay rất đa dạng và phong phú trên các hệ điều hành di động. Các hệ điều hành J2ME, Android, IOS, Hybrid, Web bases Mobile Application đã rất phát triển trên thị trường truyền thông di động.

Trong vài năm trở lại đây, hệ điều hành Android ra đời với sự kế thừa những ưu việt của các hệ điều hành ra đời trước và sự kết hợp của nhiều công nghệ tiên tiến nhất hiện nay. Android đã nhanh chóng là đối thủ cạnh tranh mạnh mẽ với các hệ điều hành trước đó và đang là hệ điều hành di động của tương lai và được nhiều người ưa chuộng nhất.

Hướng đến nhu cầu thực tế trường Đại học quản lý và công nghệ Hải phòng cần một ứng dụng trên điện thoại để đăng ký ăn trưa, đáp ứng nhanh và thuận tiên. Vì vậy em chọn đề tài ***"Xây dựng ứng dụng đăng ký ăn trưa tại trường Đại học quản lý và công nghệ Hải phòng"***

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành nhất đến quý thầy cô Trường Đại Học Quản lý và Công nghệ Hải Phòng, những người đã dìu dắt em tận tình, đã truyền đạt cho em những kiến thức và bài học quý báu trong suốt thời gian em theo học tại trường.

Em xin trân trọng gửi lời cảm ơn đến tất cả các thầy cô trong khoa Công Nghệ Thông Tin, đặc biệt là cô giáo Ths.Nguyễn Thị Xuân Hương, cô đã tận tình hướng dẫn và giúp đỡ em trong suốt quá trình làm tốt nghiệp. Với sự chỉ bảo của cô, em đã có những định hướng tốt trong việc triển khai và thực hiện các yêu cầu trong quá trình làm đồ án tốt nghiệp.

Em xin cảm ơn những người thân và gia đình đã quan tâm, động viên và luôn tạo cho em những điều kiện tốt nhất trong suốt quá trình học tập và làm tốt nghiệp.

Ngoài ra, em cũng xin gửi lời cảm ơn tới tất cả bạn bè, đặc biệt là các bạn trong lớp CT2001C đã luôn gắn bó, cùng học tập và giúp đỡ em trong những năm qua và trong suốt quá trình thực hiện đồ án này.

Em xin chân thành cảm ơn!

Hải Phòng, ngày 26 tháng 06 năm 2020

Sinh viên

Vũ Ngọc Đông

CHƯƠNG 1: GiỚI THIỆU VỀ HỆ ĐIỀU HÀNH ANDROID

1.1. Giới thiệu về hệ điều hành Android

Android là một hệ điều hành dựa trên nền tảng Linux, được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, Android được phát triển bởi Tổng công ty Android, với sự hỗ trợ tài chính từ Google và sau này được chính Google mua lại vào năm 2005.

Chính mã nguồn mở của Android cùng với tính không ràng buộc nhiều đã cho phép các nhà phát triển thiết bị di động và các lập trình viên được điều chỉnh và phân phối Android một cách tự do. Ngoài ra, Android còn có một cộng đồng lập trình viên đông đảo chuyên viết các ứng dụng để mở rộng chức năng của thiết bị.

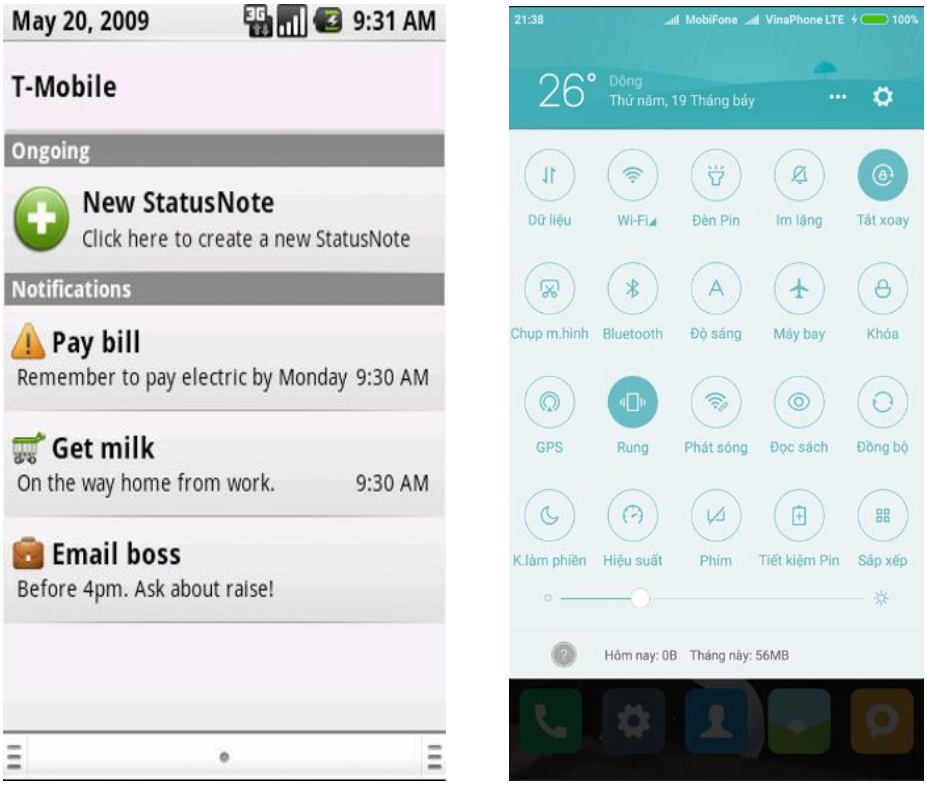
Nhờ yếu tố mở, dễ dàng tinh chỉnh cùng sự phát triển nhanh chóng đã khiến hệ điều hành này dần trở nên phổ biến, kết quả là mặc dù được thiết kế để chạy trên điện thoại và máy tính bảng nhưng giờ đây Android đã xuất hiện trên các smart TV, máy chơi game và một số thiết bị điện tử khác.

Android bắt đầu với bản beta đầu tiên vào tháng 11 năm 2007 và phiên bản thương mại đầu tiên, Android 1.0, được phát hành vào tháng 9 năm 2008. Kể từ tháng 4 năm 2009, phiên bản Android được phát triển, đặt tên theo chủ đề bánh kẹo và phát hành theo thứ tự bảng chữ cái: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, Kitkat, Lollipop, Marshmallow, Nougat, và bây giờ là Oreo.

Kỷ nguyên của Android chính thức bắt đầu vào ngày 22 tháng 10 năm 2008, khi chiếc điện thoại T-Mobile G1 bắt đầu được bán ra tại Mỹ. Vào thời gian đầu, rất nhiều tính năng cơ bản bị thiếu sót như: bàn phím ảo, cảm ứng đa điểm và tính năng mua ứng dụng vẫn chưa xuất hiện. Tuy nhiên, một số tính năng cũng như giao diện đặc sản của hệ điều hành này đã khởi nguồn từ chiếc G1 và trở thành những yếu tố không thể thiếu trên Android sau này.

Sự phát triển của hệ điều hành Android:

Thanh thông báo vuốt từ trên xuống (Notification bar): Ngay từ những ngày đầu tiên của Android, thanh thông báo này đã đánh dấu một bước quan trọng mà trước đây chưa hề có hệ điều hành nào làm được - đưa tất cả thông tin tin nhắn, tin thoại hoặc các cuộc gọi nhỡ chỉ với thao tác vuốt xuống.

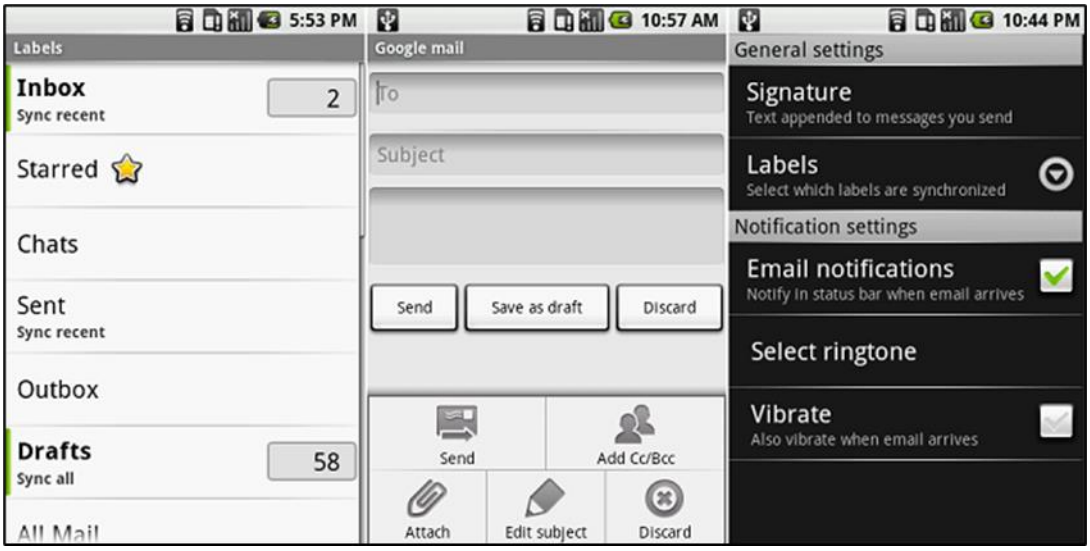


Hình 1.1.1. Thanh thông báo ở phiên bản thời kỳ đầu(trái) so với phiên bản Android 8.1(phải)

Màn hình chính (Home Screen) và các widget: Một điểm khác biệt giữa Android so với các hệ điều hành khác là phần màn hình chính của mình. Bên cạnh việc thay đổi được hình nền, Android còn cho phép người dùng tùy biến màn hình chính của mình với nhiều widgets kèm theo, chẳng hạn như đồng hồ, lịch, trình nghe nhạc, đưa các icon ứng dụng ra ngoài hoặc thậm chí có thể can thiệp sâu hơn để thay đổi toàn bộ giao diện màn hình Home Screen này.

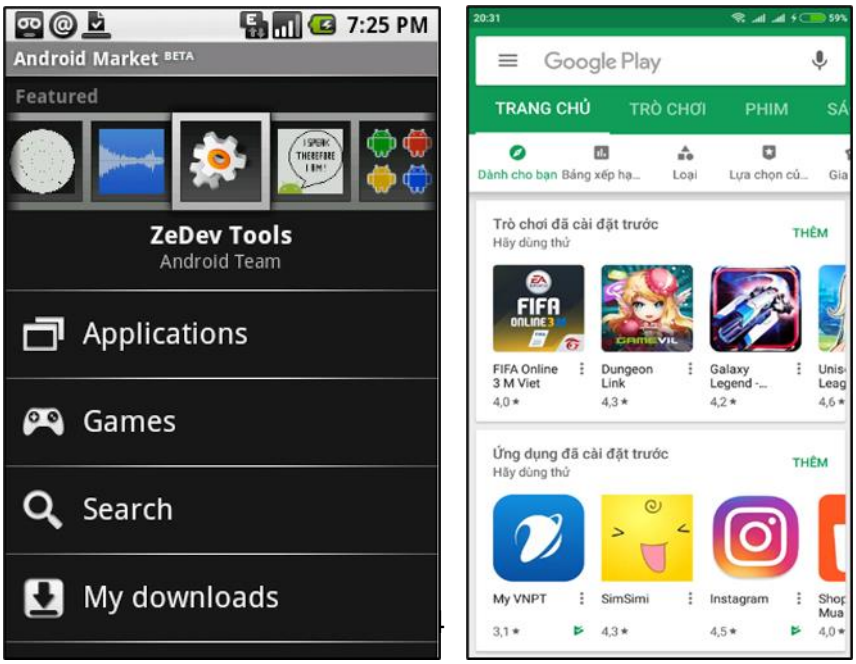
Hình 1.1.2. Màn hình chính của Android 1.0(trái) so với Android 8.1(phải)

Đồng bộ và tích hợp chặt chẽ với Gmail: Vào thời điểm điện thoại G1 được bán ra, Gmail đã hỗ trợ giao thức POP và IMAP để tích hợp với các trình email trên di động. Tuy nhiên, lúc bấy giờ không có bất kì sản phẩm nào có thể hỗ trợ được hoàn toàn những tính năng ưu việt này của Gmail. Mãi cho đến khi Android 1.0 xuất hiện, vấn đề này đã được khắc phục và G1 trở thành chiếc điện thoại mang lại trải nghiệm Gmail tốt nhất trên thị trường lúc bấy giờ.



Hình 1.1.3. Gmail trên Android thời kỳ đầu

Kho ứng dụng Android Market: Thật khó có thể tưởng tượng một chiếc smartphone mà không hề có kho ứng dụng, nhưng vào thời điểm Android mới ra mắt, gần như không có bất kì điện thoại nào có kho ứng dụng nào được tích hợp và chính Android đã mở đầu cuộc cách mạng ứng dụng di động này. Android Market trên G1 thời bấy giờ có rất ít ứng dụng và giao diện cực kỳ đơn giản, hơn nữa tính năng mua ứng dụng trên phiên bản này vẫn chưa được xuất hiện mãi cho đến năm sau - những vấn đề này dễ hiểu vì thời điểm này Android chỉ mới được khai sinh nên mọi thứ còn khá thô sơ.

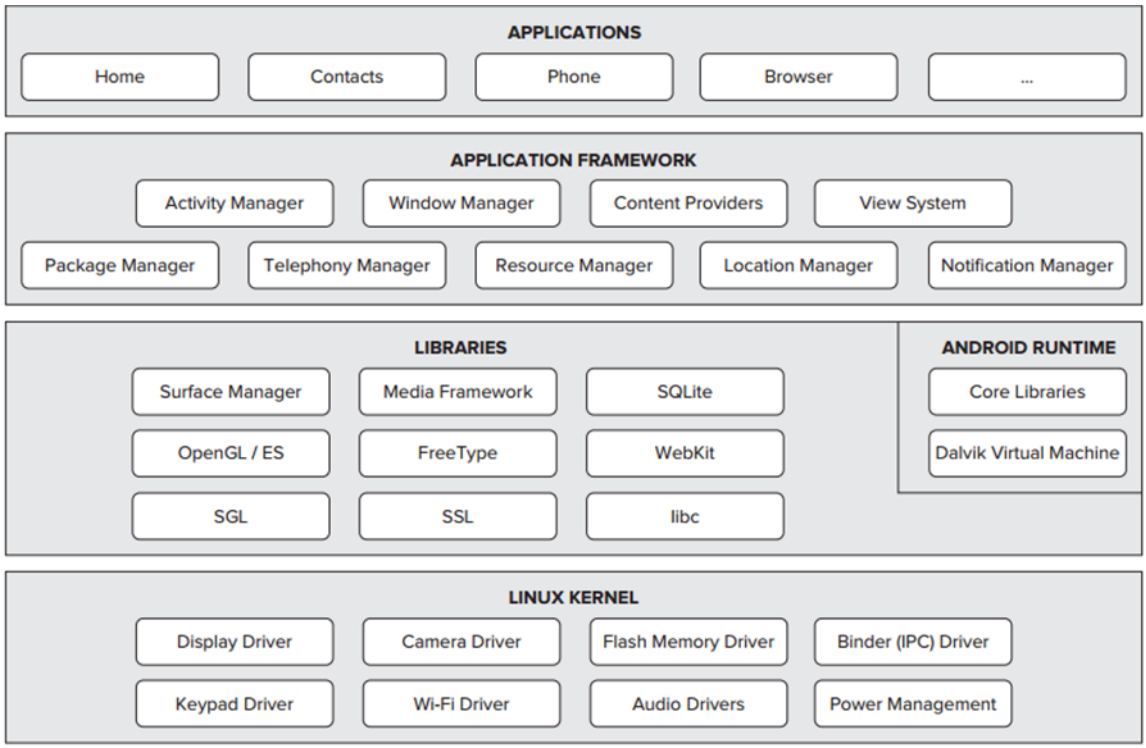


Hình 1.1.4. Hình ảnh Store của Android 1.0(trái) và Android 8.1(phải)

Giao diện: Google đã phát triển giao diện Android phiên bản 1.0 với sự hỗ trợ từ TAT, viết tắt từ The Astonishing Tribe, một công ty thiết kế tương tác của Thụy Điển. Dấu ấn rõ ràng nhất mà TAT để lại trên phiên bản Android từ phiên bản 1.0 cho đến 2.2 chính là widget đồng hồ kim nằm ngoài Home Screen tuy đơn giản nhưng rất đẹp mắt. Công ty này sau đó ngừng hợp tác với Google và bị RIM mua lại để tập trung phát triển sản phẩm Blackberry cũng như nền tảng BBX sau này.

1.2. Kiến trúc cơ bản của hệ điều hành Android

Android gồm năm phần chính sau được chứa trong bốn lớp:



H

Hình 1.2.1. Kiến trúc hệ điều hành Android

1.2.1. Nhân Linux

Android dựa trên Linux phiên bản 2.6 cho hệ thống dịch vụ cốt lõi như security, memory management, process management, network stack, and driver model. Kernel Linux hoạt động như một lớp trừu tượng hóa giữa phần cứng và phần còn lại của phần mềm stack.

1.2.2. Thư viện

Android bao gồm một tập hợp các thư viện C/C++ được sử dụng bởi nhiều thành phần khác nhau trong hệ thống Android.Điều này được thể hiện thông qua nền tảng ứng dụng Android. Một số các thư viện cơ bản được liệt kê dưới đây:

- Hệ thống thư viện C: một BSD có nguồn gốc từ hệ thống thư viện tiêu chuẩn C (libc), điều chỉnh để nhúng vào các thiết bị dựa trên Linux.
- Thư viện Media - dựa trên PacketVideo's OpenCORE; các thư viện hỗ trợ phát lại và ghi âm của âm thanh phổ biến và các định dạng video, cũng như các tập tin hình ảnh tĩnh,bao gồm cả MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG.
- Bề mặt quản lý - Quản lý việc truy xuất vào hệ thống hiển thị.
- LibWebCore - một công cụ trình duyệt web hiện đại mà quyền hạn cả hai trình duyệt web Android và xem web nhúng.
- SGL - Đồ họa 2D cơ bản của máy.
- Thư viện 3D - một thực hiện dựa vào OpenGL ES 1.0 APIs; các thư viện sử dụng phần cứng tăng tốc 3D (nếu có), tối ɑu hóa cao rasterizer phần mềm 3D.
- FreeType - vẽ phông chữ bitmap và vector.

SQLite một công cụ cơ sở dữ liệu quan hệ mạnh mẽ và nhẹ có sẵn cho tất cả các ứng dụng.

1.2.3. Thực thi

Android bao gồm một tập hợp các thư viện cơ bản mà cung cấp hầu hết các chức năng có sẵn trong các thư viện lõi của ngôn ngữ lập trình Java.Tất cả các ứng dụng Android đều chạy trong tiến trình riêng.Máy ảo Dalvik đã được viết để cho một thiết bị có thể chạy nhiều máy ảo hiệu quả.Các VM Dalvik thực thi các tập tin thực thi Dalvik (dex).Định dạng được tối ưu hóa cho bộ nhớ tối thiểu.VM là dựa trên register - based, và chạy các lớp đã được biên dịch bởi một trình biên dịch Java để chuyển đổi thành các định dạng dex.Các VM Dalvik dựa vào nhân Linux cho các chức năng cơ bản như luồng và quản lý bộ nhớ thấp.

1.2.4. Nền tảng Android

Bằng cách cung cấp một nền tảng phát triển mở, Android cung cấp 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 động, thêm các thông báo để các thanh trạng thái, và nhiều, nhiều hơn nữa.Nhà phát triển có thể truy cập vào các API cùng một khuôn khổ được sử dụng bởi các ứng dụng lõi. Các kiến trúc ứng dụng được thiết kế để đơn giản hóa việc sử dụng lại các thành phần; bất kỳ ứng dụng có thể xuất bản khả năng của và ứng dụng nào khác sau đó có thể sử dụng những khả năng (có thể hạn chế bảo mật được thực thi bởi khuôn khổ). Cơ chế này cho phép các thành phần toịong tự sẽ được thay thế bởi người sử dụng.

Cơ bản tất cả các ứng dụng là một bộ các dịch vụ và các hệ thống, bao gồm:

- Một tập hợp rất nhiều các View có khả năng kế thừa lẫn nhau dùng để thiết kế phần giao diện ứng dụng như: gridview, tableview, linearlayout
- Một “Content Provider” cho phép các ứng dụng có thể truy xuất dữ liệu từ các ứng dụng khác (chẳng hạn như Contacts) hoặc là chia sẻ dữ liệu giữa các ứng dụng đó.
- Một “Resource Manager” cung cấp truy xuất tới các tài nguyên không phải là mã nguồn, chẳng hạn như: localized strings, graphics, and layout files.
- Một “Notifycation Manager” cho phép tất cả các ứng dụng hiển thị các custom alerts trong status bar. Activity Maanager được dùng để quản lý chu trình sống của ứng dụng và điều hướng các activity.

1.2.5. Tầng ứng dụng

Tầng ứng dụng (Application) là tầng giao tiếp với người dùng với các thiết bị Android như Danh bạ, tin nhắn, trò chơi, tiện ích tính toán, trình duyệt… Mọi ứng dụng viết đều nằm trên tầng này.

CHƯƠNG 2: MÔI TRƯỜNG PHÁT TRIỂN ỨNG DỤNG ANDROID STUDIO, SQLITE

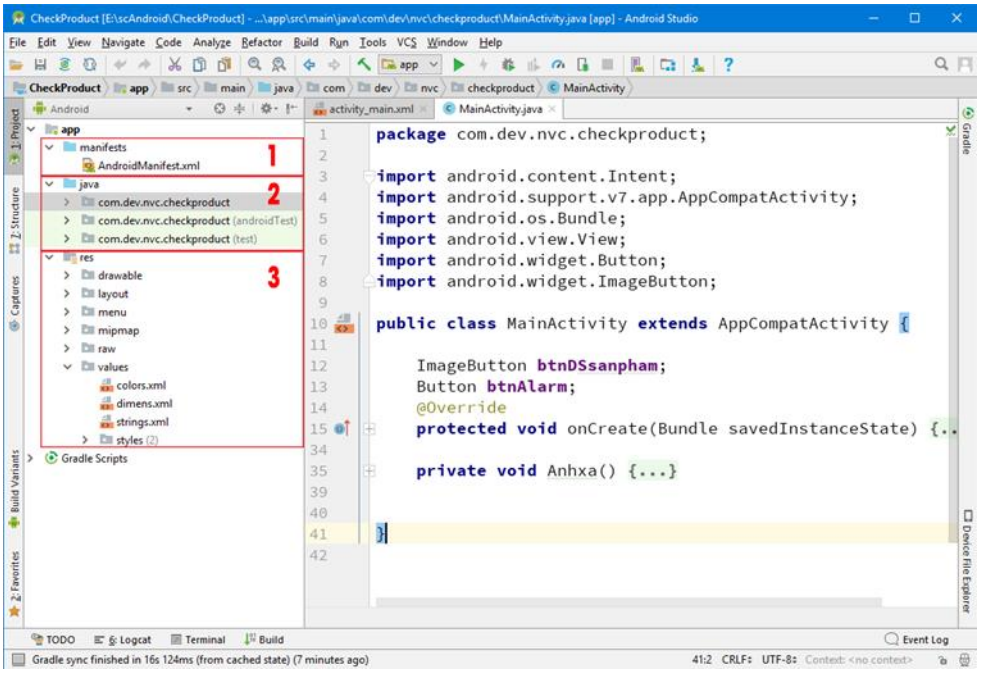
2.1. Giới thiệu ứng dụng Android Studio

Android Studio là một nền tảng IDE[1] (integrated development environment) dùng để phát triển các ứng dụng android, được Google release vào khoảng đầu năm 2015 thay thế cho bản Eclipse cũ. Android Studio được phát triển dựa trên IntelliJ IDEA Community Edition - công cụ lập trình tốt nhất cho java, giúp cho các lập trình viên tạo ứng dụng, thực hiện các thay đổi một cách dễ dàng, bên cạnh đó có thể xem trước trong thời gian thực và thiết kế giao diện đẹp hơn trước. Tiếng Việt cũng đã được tích hợp trong Android Studio. Đặc biệt, Android Studio cho phép người dùng Import Project từ Eclipse sang và logic lập trình cũng tương tự.

2.2. Thành phần trong một dự án ANDROID

Bao bọc một Project android là thư mục app và phía dưới là ba thành phần chính của một project trong Android Studio:

1



Hình 2.2.1. Cấu trúc một project trong Android Studio

2.2.1. Tệp cấu hình Android[2]

a. Giới thiệu

Trong bất kì một project Android nào khi tạo ra đều có một file AndroidManifest.xml, file này được dùng để định nghĩa các screen sử dụng, các permission cũng như các theme cho ứng dụng. Đồng thời nó cũng chứa thông tin về phiên bản SDK cũng như main activity sẽ chạy đầu tiên. File này được tự động sinh ra khi tạo một Android project. Trong file manifest bao giờ cũng có 3 thành phần chính đó là: application, permission và version. Hay nói cách khác đây là file dùng để config những thuộc tính cho ứng dụng của mà khi ứng dụng khởi chạy hệ điều hành có thể hiểu được và xử lí.

[2] AndroidManifest

b. Tác dụng của AndroidManifest

- Đặt tên gói Java cho ứng dụng. Tên gói đóng vai trò như một mã nhận diện duy nhất cho ứng dụng.
- Mô tả các thành phần của ứng dụng - hoạt động, dịch vụ, hàm nhận quảng bá, và trình cung cấp nội dung mà ứng dụng được soạn bởi. Nó đặt tên các lớp triển khai từng thành phần và công bố các khả năng của chúng (ví dụ, những tin nhắn Intent mà chúng có thể xử lý). Những khai báo này cho phép hệ thống Android biết các thành phần là gì và chúng có thể được khởi chạy trong những điều kiện nào.
- Xác định những tiến trình nào sẽ lưu trữ các thành phần ứng dụng.
- Khai báo các quyền mà ứng dụng phải có để truy cập các phần được bảo vệ của API và tương tác với các ứng dụng khác.
- Khai báo các quyền mà ứng dụng khác phải có để tương tác với các thành phần của ứng dụng.
- Liệt kê các lớp Instrumentation cung cấp tính năng tạo hồ sơ và các thông tin khác khi ứng dụng đang chạy. Những khai báo này chỉ xuất hiện trong bản kê khai khi ứng dụng đang được phát triển và thử nghiệm; chúng bị loại bỏ trước khi ứng dụng được công bố.
- Khai báo mức tối thiểu của API Android mà ứng dụng yêu cầu.
- Liệt kê các thư viện mà ứng dụng phải được liên kết với.

Cấu trúc tệp AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest>
  <uses-permission /> <permission /> <permission-tree
/> <permission-group />
  <instrumentation />
  <uses-sdk /> <uses-configuration /> <uses-feature />
  <supports-screens />
  <compatible-screens />
  <supports-gl-texture />
  <application>
    <activity>
      <intent-filter> <action /> <category /> <data /> </intent-
filter>
      <meta-data />
    </activity>
    <activity-alias>
      <intent-filter> . . . </intent-filter>
      <meta-data />
    </activity-alias>
    <service>
      <intent-filter> . . . </intent-filter>
      <meta-data/>
```

c. Một số thẻ trong AndroidManifest.xml

<uses-permission>…<uses-permission/>: đây là cú pháp lệnh để xin cấp một quyền gì đó khi ứng dụng tương tác với dữ liệu nào đó mà google không cho phép dùng tùy ý.

<activity>…</activity>: đây là thẻ để khai báo các activity trong ứng dụng và các thuộc tính của activity đó, nếu như không khai bao thì khi khởi chạy ứng dụng sẽ lỗi ngay.

<service>…</service>: đây là một dịch vụ chạy ngầm trên ứng dụng của kể cả khi ứng dụng đã tắt đi và khi sử dụng phải khai báo chúng trong AndroidMainifest.

<receiver>…</receiver>: đây là khai báo khi sử dụng Broadcast Reciver[3], cái này là để lắng nghe các sự kiện thay đổi của hệ thống như khi bất wifi, khi sạc pin…

<uses-sdk>...</uses-sdk>: Thẻ xác định phiên bản SDK

2.2.2. Thư mục Java

Đây chính là nơi chứa các gói[4] của dự án, có thể tạo các gói ở đây và bên trong là các class.

2.2.3. Thư mục Res

Gói Drawable: Đây chính là thư mục chứa các file hình ảnh, config xml… trong dự án android.Ví dụ như muốn ứng dụng sử dụng một hình ảnh nào đó là backgroud thì ảnh đó sẽ bỏ vào thư mục này. Hoặc muốn điều chỉnh một nút button khi click vào màu xanh còn khi không click vào màu trắng thì sẽ config trong file xml và lưu vào trong này.

Gói Layout: Đây chính là thư mục lưu các file xml về giao diện của các màn hình ứng dụng của. Ở trên phần số một có các package lưu các class, các class này sẽ kết nối với các file xml trong thư mục layout nào để tạo nên một màn hình có giao diện cho người dùng thao tác.

Gói Mipmap: Đây là thư mục mà sẽ chứa ảnh logo ứng dụng chúng ta, lúc này có nói là các file hình ảnh sẽ được chứa trong thư mục drawable nhưng ngoại lệ ảnh logo thì chứa trong thư mục ***mipmap*** này cho chuẩn.

Gói Values: Sẽ có rất nhiều file ở bên trong như sau:

---

[3] Một trong các lớp của Android

[4] Package

color.xml: đây là file định nghĩa các mã màu trong dự án android, khi sử dụng màu nào chỉ cần gọi tên tên mã màu đã định nghĩa trong đây ra là xong.

dimens.xml: đây là file mà sẽ định nghĩa ra các kích thước như cỡ chữ, chiều cao, chiều rộng các view.

strings.xml: đây là file định nghĩa các đoạn văn bản trong ứng dụng Android của ví dụ như có một đoạn văn bản mà sử dụng đi sử dụng lại trong các màn hình khác nhau, khi set cứng ở nhiều nơi thì khi cần chỉnh sửa thì phải tìm hết tất cả và sửa lại. Bây giờ định nghĩa đoạn văn bản đó trong đây và khi dùng thì gọi ra sử dụng và sau này chỉnh sữa chỉ cần sửa trong đây là xong, nó sẽ apply tất cả mọi nơi.

styles.xml: đây chính là nơi định nghĩa các giao diện của các file layout trong thư mục layout đã nói phía trên. Kiểu như thế này nhé, muốn chỉnh một nút Button chiều cao 10dp, chiều rộng 10dp, màu xanh… và lại sử dụng kiểu thiết kế này ở năm màn hình khác nhau. Không thể mỗi màn hình lại định nghĩa lại như thế sẽ làm duplicate code (lặp lại) và sẽ không tối ưu tí nào cả. Thay vào đó chỉ cần định nghĩa một file giao diện như trên và ở mỗi màn hình chỉ cần gọi là xong.

2.2.4. Tệp Grade Scripts

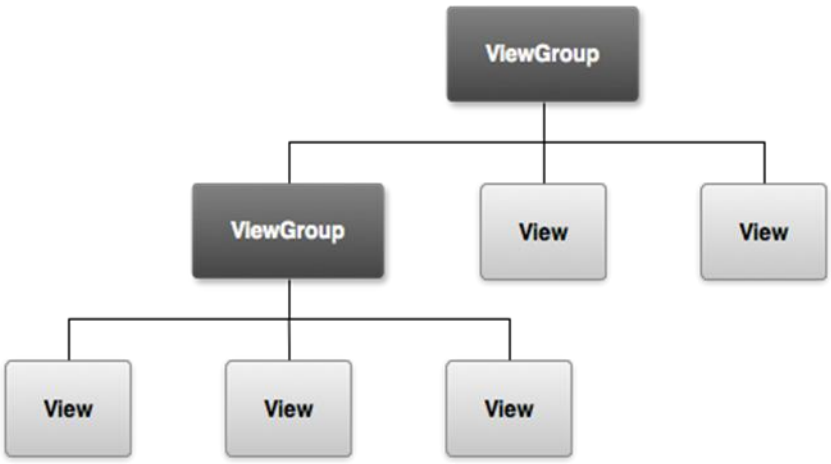
Build.grade là file để thiết lập các thuộc tính cho dự án android như: phiên bản SDK, Vesion ứng dụng, package, thêm thư viện ngoài…

2.3. Thành phần giao diện

2.3.1. View group

a. Khái niệm

Một `ViewGroup` là một đối tượng được sử dụng để chứa các đối tượng `View` và `ViewGroup` khác để tổ chức và kiểm soát layout của một màn hình. Các đối tượng `ViewGroup` được sử dụng cho việc tạo ra một hệ thống phân cấp của các đối tượng `View` (xem bên dưới) do đó có thể tạo các layout phức tạp hơn.



Hình 2.3.1.1. Sơ đồ phân cấp các thành phần giao diện

b. Một số view group cơ bản

`LinearLayout:` Tồn tại để hiển thị các phần tử theo một thứ tự xếp chồng lên nhau theo chiều ngang hoặc chiều dọc. LinearLayout cũng có thể được sử dụng để gán weight cho các phần tử `View` con để các phần tử được cách khoảng trên màn hình theo tỉ lệ tương ứng với nhau.

`RelativeLayout:` Lớp con này của `ViewGroup` cho phép hiển thị các phần tử trên màn hình tương đối với nhau, cung cấp nhiều tính linh hoạt hơn và tự do trong cách layout của xuất hiện so với `LinearLayout`.

`FrameLayout:` Được thiết kế để hiển thị một `View` con tại một thời điểm, `FrameLayout` vẽ các phần tử trong một ngăn xếp và cung cấp một cách đơn giản để hiển thị một phần tử trên các kích cỡ màn hình khác nhau.

ScrollView: Một lớp mở rộng của FrameLayout, lớp ScrollView xử lý việc cuộn các đối tượng con của nó trên màn hình.

RecyclerView: Lớp RecyclerView là một lớp con của ViewGroup, nó liên quan đến các lớp ListView và GridView và nó được cung cấp bởi Google thông qua thư viện hỗ trợ RecyclerView cho các phiên bản Android cũ hơn. Lớp RecyclerView đòi hỏi việc sử dụng các mẫu thiết kế view holder để tái sử dụng phần tử một cách có hiệu quả và nó hỗ trợ việc sử dụng một LayoutManager, một thành phần trang trí, và một phần tử động để làm cho thành phần này vô cùng linh hoạt và đơn giản.

CoordinatorLayout: Được thêm gần đây vào thư viện hỗ trợ thiết kế, lớp CoordinatorLayout sử dụng một đối tượng Behavior để xác định cách các phần tử View con sẽ được sắp xếp và di chuyển khi người dùng tương tác với ứng dụng.

2.3.2. View

a. Khái niệm

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và ViewGroup. Có nhiều kiểu View và ViewGroup. Tất cả các kiểu đó được gọi là các Widget. Tất cả mọi widget đều có chung các thuộc tính cơ bản như là cách trình bày vị trí, background, kích thước, lề,…Tất cả những thuộc tính chung này được thể hiện hết ở trong đối tượng View. Trong Android Platform, các screen luôn được bố trí theo một kiểu cấu trúc. Một screen là một tập hợp các Layout và các widget được bố trí có thứ tự.

b. Một số view cơ bản trong android

Tên widget	Chức năng
TextView	Cho phép người dùng hiển thị một đoạn văn bản lên màn hình mà không cho phép người dùng sửa nó.
EditText	Cho phép người dùng nhập, xóa, sửa một đoạn văn bản vào trong đó. Có nhiều dạng EditText khác nhau như: Plaintext, Person Name, Password, Email, Phone,…
Button	Dùng để thiết lập các sự kiện khi người dùng thao tác với nó.
ImageButton	Là dạng nút bấm nhưng có thể chèn thêm hình ảnh vào để giao diện thêm sinh động, trực quan hơn.
CheckBox	Một dạng nút bấm đặc biệt chỉ có hai trạng thái là check và uncheck.
ToggleButton	Có thể xem nó như một Checkbox và có kèm theo hiệu ứng ánh sáng bật/tắt thể hiện trạng thái Check/Uncheck.
RadioButton	Chọn một trong các Radio
Spinner	Là view thể hiện quá trình xử lý diễn ra bên dưới ứng dụng, tạo cảm giác thực cho người dùng. Có hai dạng progressBar là Large và Horizontal
ListView	Chỉ cho phép chọn một trong nhóm lựa chọn.
GridView	Là view được hiển thị dưới dạng lưới gồm nhiều item con bên trong và ta có thể tùy chỉnh nội dung cũng như các đối tượng nằm bên trong item một cách tùy ý.
ViewFlipper	Cho phép định nghĩa một tập hợp nhiều View nhưng chỉ có một View hiển thị tại một thời điểm và hỗ trợ hiệu ứng chuyển đổi giữa các View
QuickContactBage	Hiển thị hình ảnh gắn liền với một đối tượng bao gồm số điện thoại, tên, email đồng thời hỗ trợ việc gọi, nhắn tin sms, email hay tin nhắn tức thời (IM - instant message).
ImageView	Hiển thị hình ảnh
Switch	Tồn tại hai trạng thái đóng mở
ProgressBar	Thể hiện tiến trình, mức độ

2.4. Vòng đời ứng dụng android

Các Activity trong hệ thống được quản lý như một ngăn xếp activity (activity stack).

Khi một activity mới bắt đầu nó được đặt lên đầu của ngăn xếp và trở thành Running Activity (activity đang chạy), đồng thời activity trước đó sẽ nằm ngay phía dưới trong ngăn xếp đó, và sẽ không trở nên visible (nhìn thấy) cho đến khi activity ở trên thoát ra khỏi ngăn xếp.

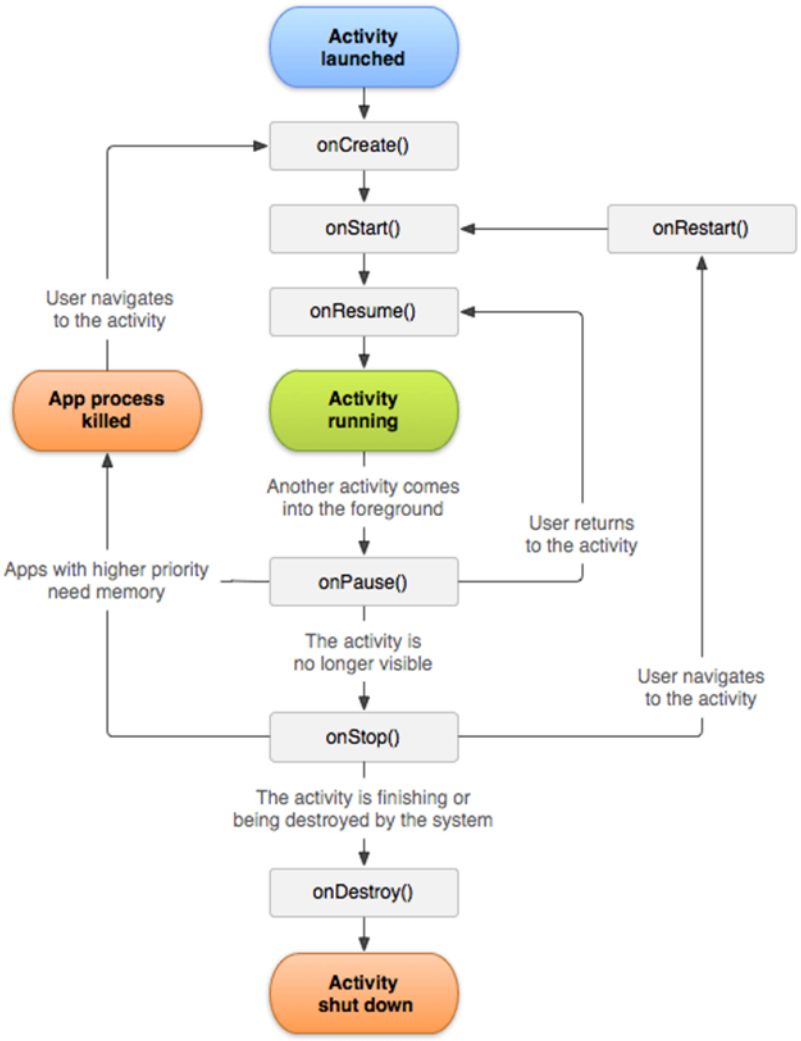
- Một Activity gồm bốn trạng thái chính:

Nếu activity ở phía trên của màn hình (hay ở trên cùng của ngăn xếp), thì nó đang ở trạng thái active (hoạt động) / running (đang chạy). Ví dụ khi ta cần gọi điện thì activity bấm số đó đang ở trạng thái active.

Nếu activity không thể tương tác nhưng vẫn nhìn thấy (khi mà bị che bởi một activity khác nhưng người dùng vẫn có thể nhìn thấy nó ở phía sau) thì activity này đang ở trạng thái paused (tạm dừng). Khi ở trạng thái này activity có thể bị xóa bỏ bởi hệ thống khi thiết bị thiếu bộ nhớ. Ví dụ khi có một activity khác dạng dialog hiện lên chỉ che đi một phần của activity hiện tại thì activity vào trạng thái paused.

Nếu activity hoàn toàn bị che khuất bởi activity khác thì nó đang ở trạng thái stopped (đã dừng). Activity này vẫn giữ được tất cả trạng thái và thông tin, nhưng không còn hiển thị với người dùng và thường xuyên bị xóa bỏ bởi hệ thống khi thiếu bộ nhớ. Ví dụ khi ta tắt màn hình thì khi đó activity vào trạng thái stopped.

Nếu activity ở trạng thái paused (tạm dừng) hay stopped (đã dừng), hệ thống có thể xóa bỏ activity đó khỏi bộ nhớ bằng cách yêu cầu nó tự kết thúc hoặc xóa bỏ tiến trình của nó. Khi acitivty đó hiển thị lại với người dùng thì sẽ được khởi tạo lại và khôi phục lại trạng thái trước đó.



Hình 2.4.1. Vòng đời của một Activity

2.5. Lớp Intent

Intent là một thành phần quan trọng trong android. **Intent** cho phép các thành phần ứng dụng có thể yêu cầu các hàm từ các thành phần ứng dụng android khác.

Intent là đối tượng của lớp **android.content.I ntent**. Mã của có thể gửi **Intent** vào hệ thống **Android** với chỉ định thành phần mục tiêu gửi đến.

Một đối tượng **Intent** có thể chứa dữ liệu thông qua một đối tượng của lớp **Bundle**. Dữ liệu này có thể được sử dụng bởi các thành phần tiếp nhận.

2.6. Share prefenrences

Đây là một class **Interface** cho phép lưu trữ và đọc dữ liệu với bằng các cặp key và value và nó được lưu dưới dạng một file xml, dữ liệu nó có thể lưu là ở dạng nguyên thuỷ như: int, float, string, boolean, long. Dữ liệu của Shared Preferences sẽ được lưu ở trong ứng dụng android luôn chính vì thế nếu các xoá ứng dụng đi hoặc là xoá dữ liệu app thì dữ liệu này sẽ hoàn toàn bị biến mất.

2.7. Hiệu ứng trong android

Hiệu ứng cơ bản

a. Fade in: Là hiệu ứng làm mờ đối tượng.

b. Fade out: Là hiệu ứng làm mờ dần đối tượng.

c. Blink: Là hiệu ứng làm nhấp nháy đối tượng.

d. Zoom in: Là hiệu ứng phóng to đối tượng.

e. Zoom out: Là hiệu ứng thu nhỏ đối tượng.

f. Rotate: Là hiệu ứng xoay đối tượng.

g. Move: Là hiệu ứng dịch chuyển đối tượng.

h. Side up: Là hiệu ứng trượt đối tượng lên trên.

i. Side down: Là hiệu ứng trượt đối tượng xuống dưới.

j. Sequential animation: Lặp lại một trong các hiệu ứng trên.

k. Together animition: Xảy ra đồng thời hai hoặc nhiều trong các hiệu ứng trên.

2.8. SQLite

2.8.1. Giới thiệu SQLite

SQLite là một cơ sở dữ liệu quan hệ, mã nguồn mở và được tích hợp sẵn trên Android.

SQLite thường được sử dụng trong các ứng dụng ở Local, như các ứng dụng Danh bạ, Tin nhắn, Ghi chú, Quản lý thông tin cá nhân, Các tùy chọn thiết lập (Setting) trong phần mềm,…etc…

2.8.2 Cấu hình SQLite

NHÓM TRUY VẤN KHÔNG TRẢ VỀ DỮ LIỆU:

– Tạo một bảng: Create Ten_bang (Danh_sach_cac_cot Kieu_du_lieu(Kich_thuoc)).

– Xóa một bảng: Drop Table Ten_bang.

– Thêm một dòng vào bảng: Insert Into Ten_bang(Cac_cot) Values (Gia_tri_tuong_ung).

– Sửa một dòng trong bảng: Update Ten_bang Set Gia_tri Where Dieu_kien.

– Xóa một dòng trong bảng: Delete From Ten_bang Where Dieu_kien.

NHÓM TRUY VẤN TRẢ VỀ DỮ LIỆU:

– Lấy ra các dòng trong bảng: Select Ten_cot From Ten_bang Where Dieu_kien.

– Lấy ra tất cả dòng trong bảng: Select * From Ten_bang.

Tạo một project mới trong Android Studio, tạo thêm một Java Class mới và đặt tên là DataSqlite. Sau đó cho nó extends từ SQLiteOpenHelper. Lúc này Class sẽ báo lỗi do thiếu một vài phương thức, lần lượt tạo một Constructor và 2 phương thức onCreate và onUpgrade cho Class để hết lỗi.

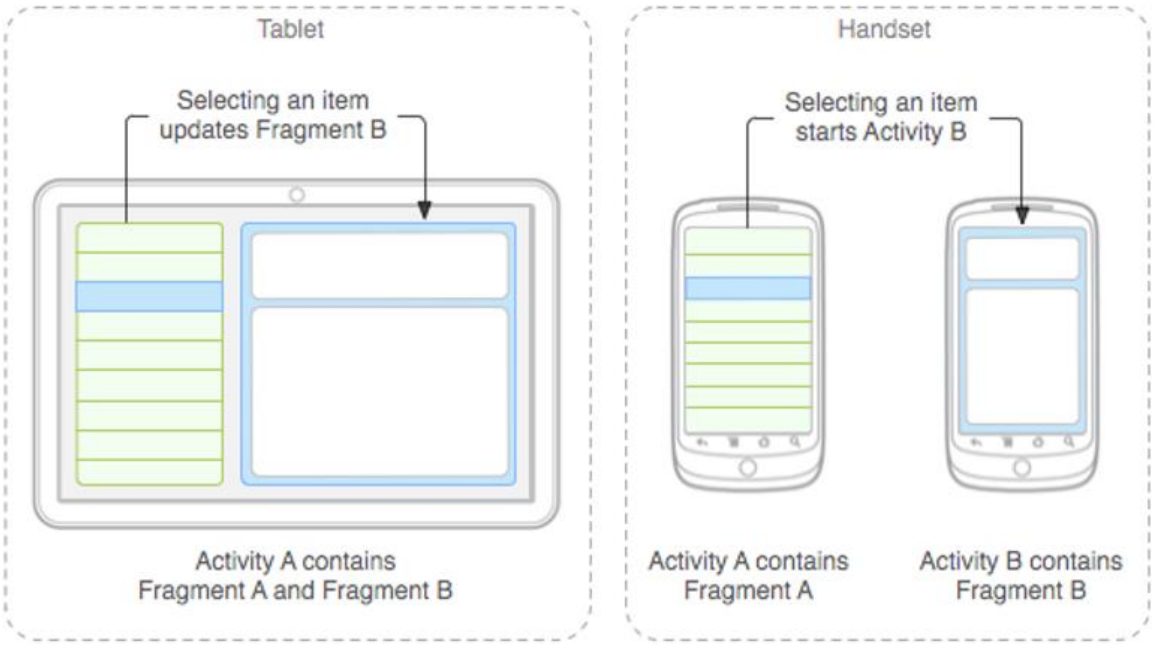
2.9. Fragment

2.9.1. Tổng quan

a. Lý do ra đời của fragment

Lý do cần phải tạo ra fragment?

- Trước đây, trên Honeycomb, Android giới thiệu khái niệm Fragments. ta có thể xem nó như là các khối được xây dựng riêng biệt với vòng đời riêng trong một Activity. Nó hỗ trợ rất nhiều trong việc **tối ưu cho các loại màn hình**, đồng thời dễ dàng được quản lý bởi activity cha, có thể **sử dụng lại, kết hợp và bố trí theo ý muốn**.
- Việc chạy từng activity riêng cho mỗi màn hình ứng dụng sẽ có hiệu quả rất tệ khi hệ thống phải cố lưu trữ chúng trong bộ nhớ lâu hết mức có thể. Tắt một cái trong số đó cũng không giải phóng các tài nguyên được sử dụng bởi những cái còn lại.



b. Vòng đời của fragment

- Fragment là một thành phần android độc lập có vòng đời và giao diện riêng được quản lý bởi một activity và hoạt động giống như một sub-activity
- Vòng đời của fragment bị ảnh hưởng trực tiếp bởi vòng đời của activity chứa nó. Tức là, khi activity bị tạm dừng thì tất cả các fragment được chứa bởi activity đó cũng tạm dừng, và khi activity bị hủy thì tất cả các fragment bên trong cũng bị hủy theo.
- Khi Fragment được gắn vào Activity, các callback **onAttach(), onCreate(), onCreateView(), onActivityCreated(), onStart(), onResume()** lần lượt được gọi.
- Sau khi các callback trên được gọi, fragment lúc đó mới chính thức được xem là đang chạy.

Sau đó, nếu người dùng bấm nút **Back** hay có bất kì thao tác gỡ/ thay thế fragment ra khỏi activity nào thì các callback **onPause(), onStop(), onDestroyView(), onDestroy(), onDetach()** sẽ được gọi. (Đây là trường hợp fragment chưa được thêm vào back stack. Ở phía dưới mình sẽ nói về trường hợp fragment được thêm vào back stack sau).

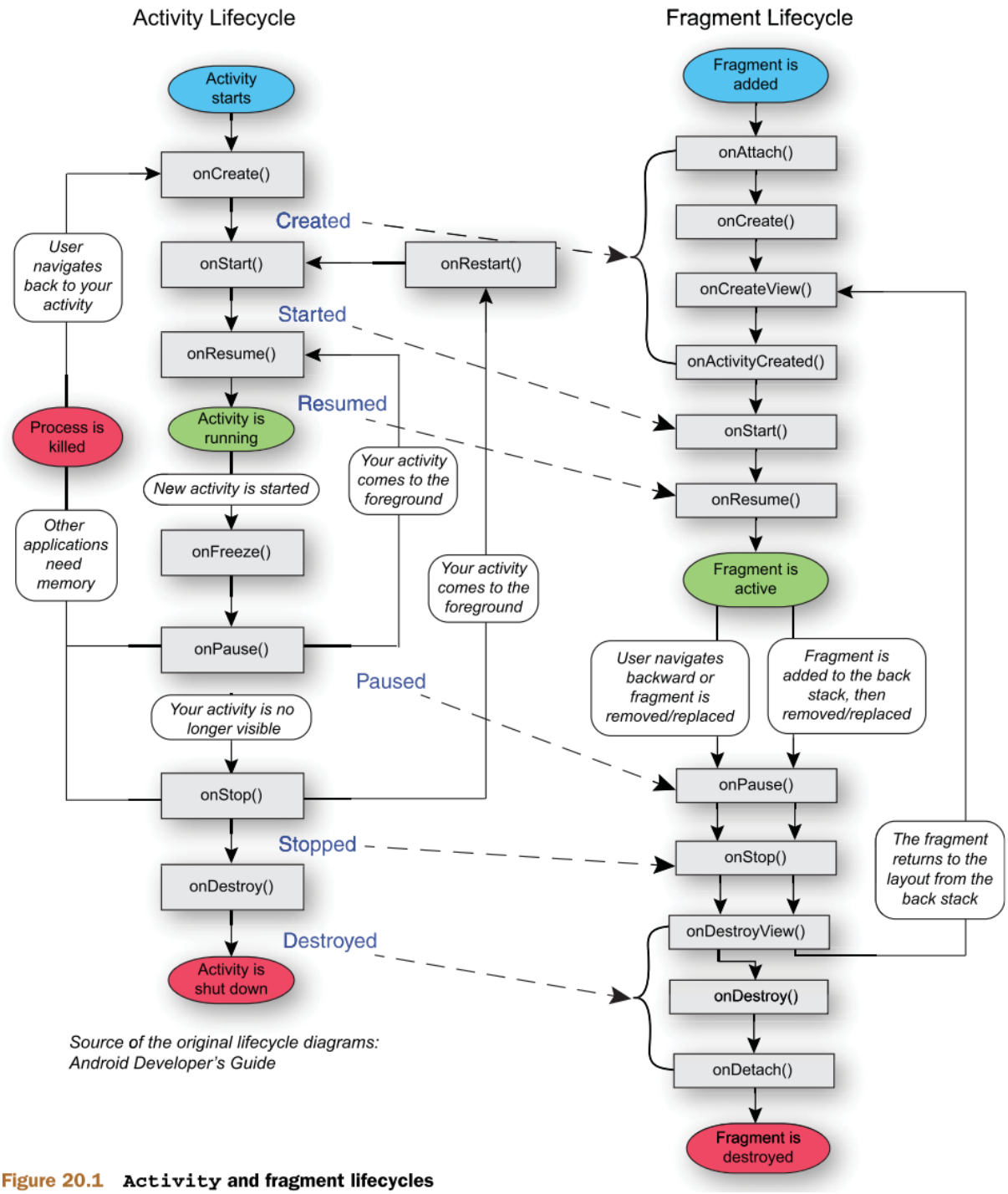


Figure 20.1 **`Activity` and fragment lifecycles**

Hình 29.2. Vòng đời của Activity và

Như vậy, việc quản lý vòng đời của một fragment rất giống với quản lý vòng đời của một activity. Giống như activity, fragment có thể tồn tại ở **3 trạng thái**:

Hoạt động (Resume): Khi fragment được gắn vào activity, có thể nhìn thấy và có thể tương tác được.

Tạm dừng (Pause): Nếu activity chứa fragment bị che lấp bởi 1 activity khác nhưng không bị che hoàn toàn, người dùng vẫn nhìn thấy được activity bị che lấp, chỉ là không tương tác được thì cả activity và fragment đều đi vào trạng thái tạm dừng.

Dừng (Stop): Cũng giống như activity, fragment bị dừng khi bị thành phần nào đó che mất hoàn toàn. Ở trạng giai đoạn, các trạng thái của của fragment vẫn được giữ lại phòng trường hợp fragment được hiển thị trở lại. Và nếu nó không còn được hiển thị với người dùng thì fragment sẽ bị gỡ bỏ nếu activity bị hủy.

2.9.2. Sử dụng

1. Tạo và hiển thị fragment

a. Hiển thị kiểu tĩnh

Đay là cách thực hiện rất nhanh chóng. Người lập trình chỉ cần sử dụng một layout có tên là fragment để hiển thị một fragment mà bạn mong muốn. Layout fragment này cũng cần bạn chỉ định các thuộc tính android:layout_width và android:layout_height như các layout khác. Chính vì vậy bạn có thể thiết kế bao nhiêu fragment vào trong giao diện của Activity đều được, và đặt chúng vào vào bất cứ vị trí nào bạn muốn. Chính thuộc tính android:name của thẻ fragment này sẽ giúp bạn chỉ định fragment nào cần hiển thị.

b. Hiển thị theo kiểu động

Nếu như với cách hiển thị tĩnh trên kia, bạn phải chỉ định thẻ fragment nào sẽ chứa đựng Fragment nào một cách cố định. Thì với cách hiển thị động này, bạn chỉ cần khai báo một vùng không gian nào đó sẽ chứa đựng fragment, vùng không gian đó được khai báo bằng một FrameLayout.

2.9.3. Tổng kết

Fragment là một thành phần android độc lập, được sử dụng bởi một activity, giống như một sub-activity.

Fragment có vòng đời và giao diện riêng.

Các Fragment thường có một file java đi kèm với file giao diện xml. Các fragment không có file giao diện xml thường được gọi là headless fragments.

Vòng đời của fragment bị ảnh hưởng trực tiếp bởi vòng đời của activity chủ. Ví dụ, khi hoạt động bị tạm dừng, tất cả phân đoạn trong nó cũng vậy, và khi hoạt động bị hủy, tất cả phân đoạn cũng vậy.

Một Fragment có thể được sử dụng trong nhiều Activitiy.

Fragment được thêm vào API 11 trở lên.

Fragment sử dụng phương thức getActivity() để lấy ra Activity cha

Fragment được định nghĩa trong file xml của activity (static definition) hoặc có thể sửa đổi fragment khi đang chạy (dynamic definition)

2.10. Tìm hiểu về DAO và DTO

2.10.1. DAO

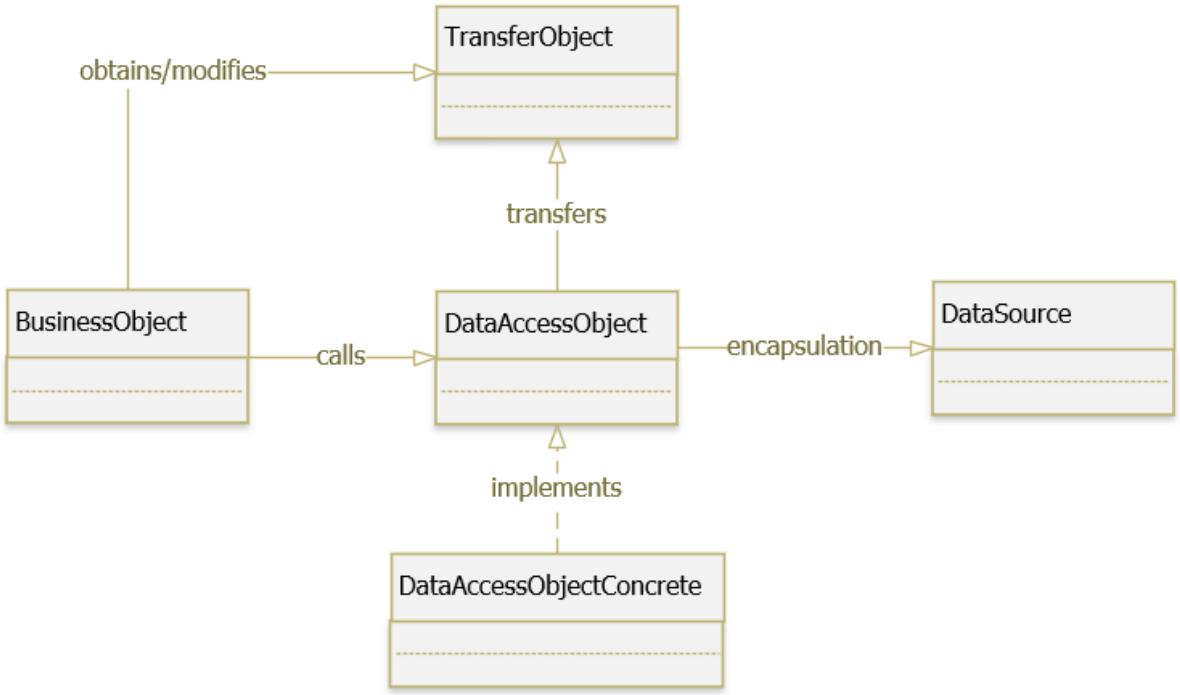
a.Khái niệm

Data Access Object (DAO) Pattern là một trong những Pattern thuộc nhóm cấu trúc (Structural Pattern). Mẫu thiết kế DAO được sử dụng để phân tách logic lưu trữ dữ liệu trong một lớp riêng biệt. Theo cách này, các service được che dấu về cách các hoạt động cấp thấp để truy cập cơ sở dữ liệu được thực hiện. Nó còn được gọi là nguyên tắc Tách logic (**Separation of Logic**).

Ý tưởng là thay vì có logic giao tiếp trực tiếp với cơ sở dữ liệu, hệ thống file, dịch vụ web hoặc bất kỳ cơ chế lưu trữ nào mà ứng dụng cần sử dụng, chúng ta sẽ để logic này sẽ giao tiếp với lớp trung gian DAO. Lớp DAO này sau đó giao tiếp với hệ thống lưu trữ, hệ quản trị CSDL như thực hiện các công việc liên quan đến lưu trữ và truy vấn dữ liệu (tìm kiếm, thêm, xóa, sửa…).

DAO dựa trên nghuyên tắc thiết kế **abstraction** và **encapsulation.** Nó bảo vệ phần còn lại của ứng dụng khỏi mọi thay đổi trong lớp lưu trữ, ví dụ: thay đổi database từ Oracle sang MySQL, thay đổi công nghệ lưu trữ từ file sang database.

Trong Java, DAO được triển khai theo nhiều cách khác nhau như Java Persistence API, Enterprise Java Bean (EJP), Object-relational mapping (ORM) với các implement cụ thể như Hibernate, iBATIS, Spring JPA, …



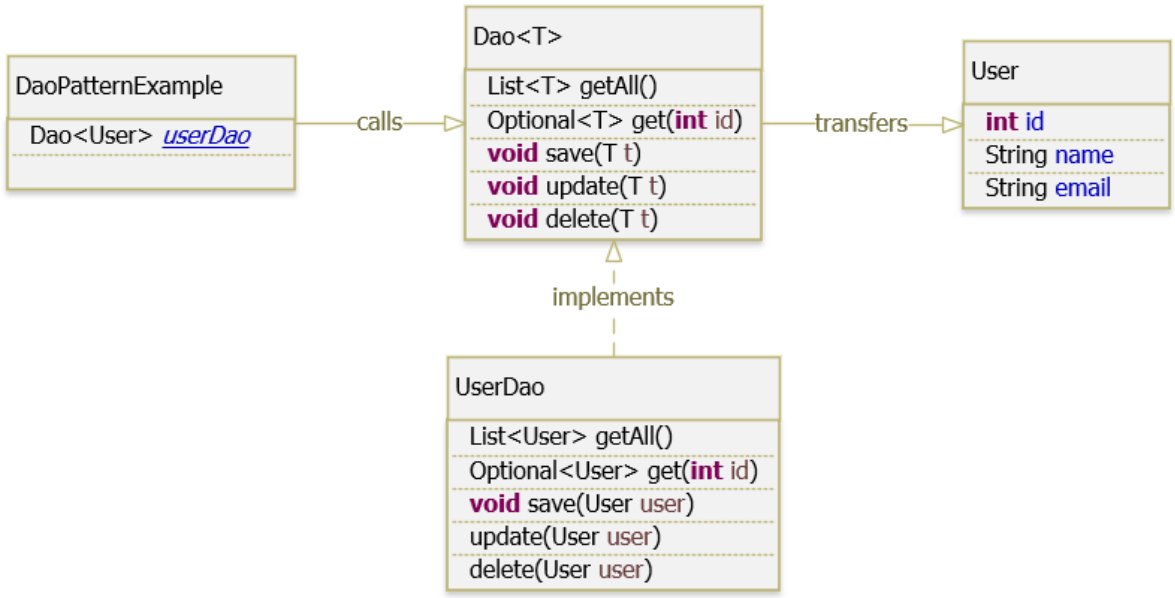
b.Cài đặt

Các thành phần tham gia mẫu Data Access Object (DAO) Pattern:

- **BusinessObject**: đại diện cho Client, yêu cầu truy cập vào nguồn dữ liệu để lấy và lưu trữ dữ liệu.
- **DataAccessObject** (DAO): là một interface định nghĩa các phương thức trừu tượng việc triển khai truy cập dữ liệu cơ bản cho BusinessObject để cho phép truy cập vào nguồn dữ liệu (DataSource).
- **DataAccessObjectConcrete**: cài đặt các phương thức được định nghĩa trong DAO, lớp này sẽ thao tác trực tiếp với nguồn dữ liệu (DataSource).
- **DataSource**: là nơi chứa dữ liệu, nó có thể là database, xml, json, text file, webservice, …

- **TransferObject**: là một POJO (Plain old Java object) object, chứa các phương thức get/set được sử dụng để lưu trữ dữ liệu và được sử dụng trong DAO class.

Ví dụ sử dụng DAO Pattern



User.java

```
1   package com.gpcoder.patterns.structural.dao;
2
3   import lombok.AllArgsConstructor;
4   import lombok.Data;
5
6   @Data
7   @AllArgsConstructor
8   public class User {
9
      private int id;
```

```
10          private String name;
11          private String email;
12      }
13
14
15
```

Dao.java

```
1   package com.gpcoder.patterns.structural.dao;
2   import java.util.List;
3   import java.util.Optional;
4   public interface Dao<T> {
5       List<T> getAll();
6       Optional<T> get(int id);
7       void save(T t);
8       void update(T t);
9       void delete(T t);
    }
```

UserDao.java

```
package com.gpcoder.patterns.structural.dao;
import java.util.ArrayList;
import java.util.List;
```

```
import java.util.Optional;
public class UserDao implements Dao<User> {
    private List<User> users = new ArrayList<>();
    public UserDao() {
        users.add(new User(1, "GP Coder", "contact@gpcoder.com"));
        users.add(new User(2, "Giang Phan", "gpcodervn@gmail.com"));
    }
    @Override
    public List<User> getAll() {
        return users;
    }

    @Override
    public Optional<User> get(int id) {
        return users.stream().filter(u -> u.getId() == id).findFirst();
    }

    @Override
    public void save(User user) {
        users.add(user);
    }

    @Override
    public void update(User user) {
        get(user.getId()).ifPresent(existUser -> {
            existUser.setName(user.getName());
            existUser.setEmail(user.getEmail());
        });
    }
```

```
    @Override
    public void delete(User user) {
        get(user.getId()).ifPresent(existUser -> users.remove(existUser));
    }
}
```

User.java

```
1       user1: User(id=1, name=GP Coder, email=contact@gpcoder.com)
2       All users:
3       User(id=1, name=updated.GP Coder, email=contact@gpcoder.com)
4       User(id=2, name=Giang Phan, email=gpcodervn@gmail.com)
```

Lợi ích của DAO Pattern

Giảm sự kết nối (loose coupling) giữa logic nghiệp vụ (Business) và logic lưu trữ (Persistence).

Mẫu DAO cho phép đóng gói code để thực hiện thao tác CRUD, ngăn chặn việc implement riêng lẻ trong từng phần khác nhau của ứng dụng.

Dễ mở rộng, bảo trì: tất cả các chi tiết lưu trữ được ẩn khỏi phần còn lại của ứng dụng. Do đó, những thay đổi có thể được thực hiện bằng cách chỉ sửa đổi một implement của DAO trong khi phần còn lại của ứng dụng không bị ảnh hưởng. DAO hoạt động như một trung gian giữa ứng dụng và cơ sở dữ liệu.

Dễ hiểu: mọi người đều theo một quy chuẩn đã được định sẵn, nên dễ hiểu hơn, tiết kiệm được nhiều thời gian hơn.

Trong một dự án lớn hơn, các nhóm khác nhau làm việc trên các phần khác nhau của ứng dụng, mẫu DAO cho phép phân tách rõ ràng các thành phần này.

Sử dụng DAO Pattern:

Khi muốn thay đổi nguồn dữ liệu sau này, như chuyển từ cơ dữ liệu MySQL sang Oracle, SQL Server, …

Khi muốn phân tách rõ ràng các thành phần của ứng dụng.

2.10.2. DTO

a.Transfer Object Pattern:

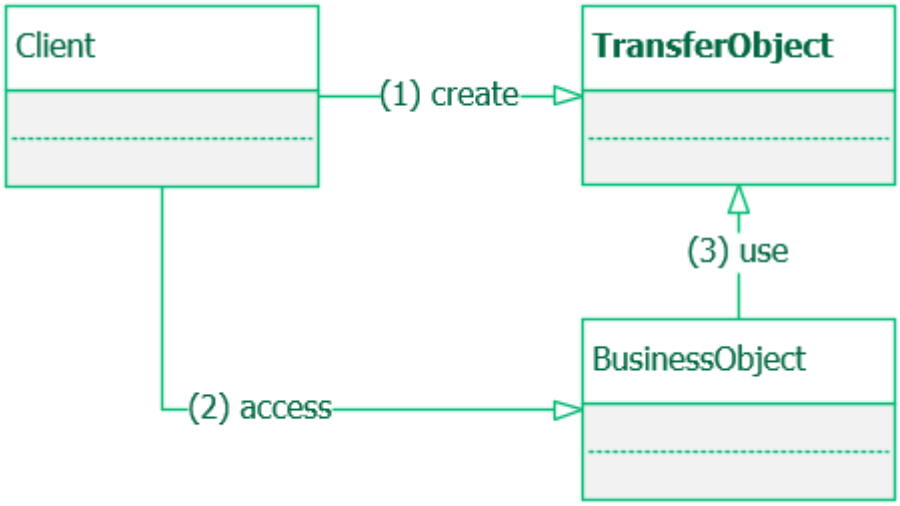
Transfer Object/ Data Transfer Object Pattern là một dạng Architectural Design Pattern, được sử dụng khi chúng ta muốn truyền dữ liệu qua lại giữa các tầng trong ứng dụng, giữa Client – Server. Data Transfer Object (DTO) còn được gọi là **Value Object** (VO).

Transfer Object đơn giản là một POJO (Plain Old Java Object), chỉ chứa các getter/ setter method và có thể có implement serialize để truyền tải dữ liệu thông qua network.

DTO hoàn toàn không chứa behavior/ logic, chỉ được sử dụng để truyền dữ liệu và map dữ liệu từ các Domain Model trước khi truyền tới Client. Trong các ứng dụng đơn giản, các Domain Model thường có thể được sử dụng lại trực tiếp dưới dạng DTO và được truyền trực tiếp đến lớp hiển thị, do đó chỉ có một Data Model thống nhất.

Đối với các ứng dụng phức tạp hơn, chúng ta không muốn hiển thị toàn bộ Domain Model cho Client, do đó, việc ánh xạ từ các Domain Model sang DTO là cần thiết.

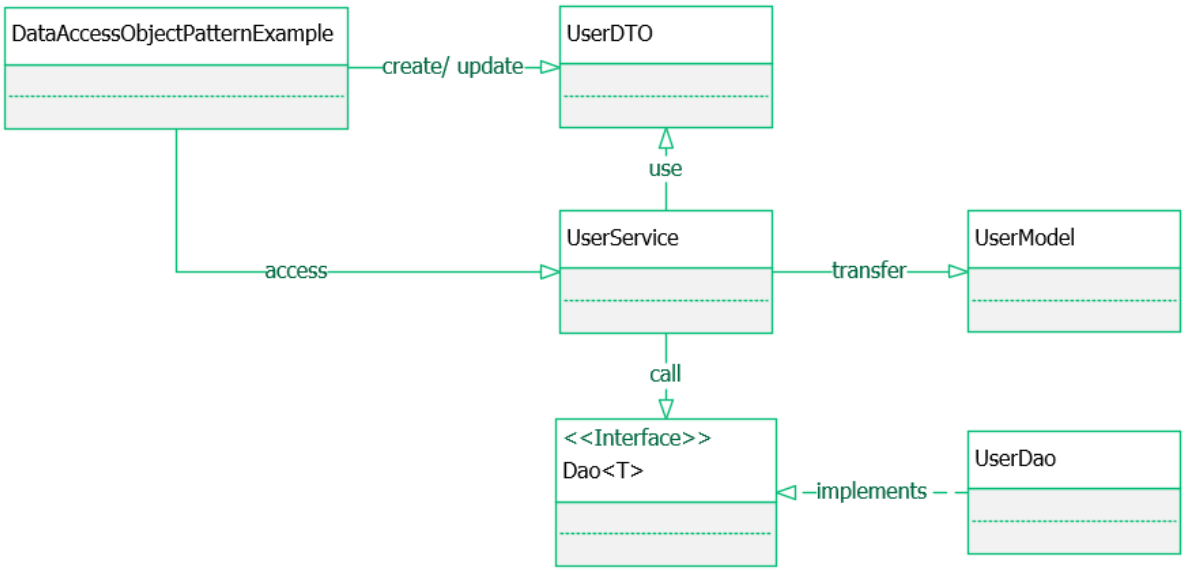
b.Cài đặt Transfer Object Pattern



Các thành phần tham gia Transfer Object Pattern:

- **Business Object**: là một Business Service, tạo Transfer Object và trả nó về Client khi cần thiết. Nó cũng có thể nhận dữ liệu từ Client trong một Transfer Object và gửi đến Server để cập nhật vào database.
- **Transfer Object**: là một POJO, chỉ chứa các getter/ setter method.
- **Client**: người sử dụng ứng dụng.
- Ví dụ sử dụng Transfer Object Pattern

Lớp xử lý nghiệp vụ ở phía Server thường truy vấn dữ liệu từ database và gán các giá trị vào Transfer Object để gửi lại Client. Phía Client có thể tạo một Transfer Object và gán giá trị vào để gửi lại Server thực hiện update vào database.



c. Lợi ích của Transfer Object Pattern

- Tách biệt logic một cách rõ ràng: Transfer Object chỉ chứa data, còn logic được implement trong phần khác.
- Cải thiện hiệu suất ứng dụng: chi phí của mỗi request/ response là lớn, chúng ta nên cố gắng gửi nhiều nhất có thể. Để làm điều này, chúng ta có thể tạo một Transfer Object để gửi data từ Client lên Server hay từ Server đến Client một lần duy nhất, thay vì phải gửi từng phần riêng lẻ.
- Giảm kết dính giữa các tầng trong ứng dụng: Client chỉ thao tác với Transfer Object, nên nó không bị ảnh hưởng khi Domain Model thay đổi.
- Bao đóng các đối số: một phương thức có nhiều đối số, chúng ta có thể bao đóng chúng trong một Transfer Object. Giúp chúng ta dễ dàng mở rộng, thêm/ bớt đối số.
- Nhận nhiều dữ liệu trả về: trong Java, một phương thức chỉ có thể trả về một giá trị, để có thể nhận được nhiều giá trị, chúng ta có thể bao đóng chúng trong một Transfer Object.
- Tăng bảo mật ứng dụng: tùy vào người dùng khác nhau có thể xem được một số dữ liệu nhất định. Chúng ta có thể tạo nhiều Transfer Object khác nhau cho từng loại người dùng thay vì trả về một Domain Object một cách trực tiếp. Trường hợp rõ ràng nhất là User Model, domain object này chứa thông tin cả email,

password, số tài khoản ngân hàng. Chúng ta có thể tạo một Transfer Object đơn giản chỉ chứa thông tin họ tên, ngày sinh. Không cần thiết phải trả tất cả dữ liệu Domain Model về Client.

- Transfer Object thường được sử dụng với Data Access Object.

2.11. Dịch vụ web

2.11.1 Khái niệm

Dịch vụ web (Web Services) là những thành phần ứng dụng dùng để chuyển đổi một ứng dụng thông thường sang một ứng dụng web. Đồng thời nó cũng xuất bản các chức năng của mình để mọi người dùng internet trên thế giới đều có thể sử dụng thông qua nền tảng web. Web Services truyền thông tin bằng cách sử dụng các giao thức mở, tài nguyên phần mềm có thể xác định bằng địa chỉ URL, thực hiện các chức năng và đưa ra các thông tin người dùng yêu cầu, các ứng dụng độc lập và tự mô tả chính nó. Nó bao gồm các modun độc lập cho hoạt động của khách hàng và doanh nghiệp và bản thân nó được thực thi trên server. Bất cứ một ứng dụng nào cũng đều có thể có một thành phần WS. WS có thể được tạo ra bằng bất kỳ một ngôn ngữ lập trình nào

2.11.2 Đặc điểm

- Cho phép client và server tương tác ngay cả trong môi trường khác nhau. (Ví dụ server chạy linux, client chạy windows).
- Phần lớn được xây dựng dựa trên mã nguồn mở và phát triển các chuẩn đã được công nhận.
- Nó có thể triển khai bởi 1 phần mềm ứng dụng phía server (Ví dụ : PHP, Oracle Application server, Microsoft .NET)

2.11.3 Những chuẩn chính của Web Services

- SOAP (Simple Object Access Protocol)
- WSDL (Web Service Description Language)
- UDDI (Universal Description, Discovery, and Integration)
- REST (Representational State Tranfer)

SOAP?

SOAP là một giao thức giao tiếp có cấu trúc như XML và mã hóa thành định dạng chung cho các ứng dụng trao đổi với nhau. Ý tưởng bắt đầu từ

Microsoft và phần mềm Userland. Một đặc tả việc sử dụng các tài liệu XML theo dạng các thông điệp.

Đặc tả về SOAP định nghĩa một mô hình trao đổi dữ liệu dựa trên 3 khái niệm cơ bản: Các thông điệp là các tài liệu XML, chúng được truyền đi từ bên gửi đến bên nhận, bên nhận có thể chuyển tiếp dữ liệu đến nơi khác.

Đặc trưng của SOAP

• SOAP được thiết kế đơn giản và dễ mở rộng.
• Tất cả các message SOAP đều được mã hóa sử dụng XML.
• SOAP sử dùng giao thức truyền dữ liệu riêng.
• Không có garbage collection phân tán, và cũng không có cơ chế tham chiếu. Vì thế SOAP client không giữ bất kỳ một tham chiếu đầy đủ nào về các đối tượng ở xa.
• SOAP không bị ràng buộc bởi bất kỳ ngôn ngữ lập trình nào hoặc công nghệ nào.

WSDL?

• WSDL là một ngôn ngữ dựa trên nền XML dùng để định vị và mô tả WS.
• WSDL là viết tắt của Web services Description language.
• WSDL dựa trên nền tảng XML.
• WSDL được dùng để mô tả WS.

Nhưng những thiệt hại lớn sẽ xảy ra vào khoảng thời gian chết của dịch vụ web, có quá nhiều chuẩn cho dịch vụ web khiến người dùng khó nắm bắt, phải quan tâm nhiều hơn tới vấn đề an toàn và bảo mật.

• WSDL được dùng để định vị WS.
• WSDL là một tiêu chuẩn W3C.

WSDL định nghĩa cách mô tả web service theo cú pháp tổng quát XML, bao gồm các thông tin:

• Tên service
• Giao thức và kiểu mã hóa sẽ được sử dụng khi gọi các hàm của web service.
• Loại thông tin: những thao tác, những tham số, những kiểu dữ liệu gồm có giao diện của web service, công với tên cho giao diện này.
• Phần giao diện mô tả giao diện và giao thức kết nối, phần thi hành mô tả thông tin để truy xuất service.

UDDI?

• UDDI là một thư mục dịch vụ, nơi mà chúng ta có thể đăng ký và tìm kiếm các dịch vụ web.

• UDDI là viết tắt của Universal Description, Discovery and integration.
• UDDI là một thư mục dùng để lưu trữ thông tin về các dịch vụ web.
• UDDI là một thư mục các giao diện dịch vụ web được mô tả bởi WSDL.
• UDDI giao tiếp thông qua SOAP.
• UDDI được xây dựng để góp phần vào nền tảng Microsoft .NET.

Để có thể sử dụng các dịch vụ, trước tiên client phải tìm dịch vụ, ghi nhận thông tin về cách sử dụng dịch vụ và biết được đối tượng cung cấp dịch vụ. UDDI định nghĩa một số thành phần cho biết trước các thông tin này để cho phép các client truy tìm và nhận lại những thông tin yêu cầu sử dụng web services.

Cấu trúc UDDI gồm các thành phần:
• White pages: chứa thông tin liên hệ và các định dạng chính yếu của web services (tên giao dịch, địa chỉ...) Những thông tin này cho phép các đối tượng khác xác định được service.
• Yellow pages: chứa thông tin mô tả web service theo những chủng loại khác nhau. Những thông tin này cho phép các đối tượng thấy web service theo từng chủng loại của nó.
• Green pages: chứa thông tin kỹ thuật mô tả các hành vi và các chức năng của web service để tìm kiếm.

REST?

REST (Representational State Tranfer) là một kiến trúc phần mềm cho các hệ thống phân tán siêu truyền thông như WWW, được chọn sử dụng rộng rãi thay cho Web service dựa trên SOAP và WSDL.

Đặc trưng của REST
• Là dạng client – server.
• Phân tách giao diện của client ra khỏi dữ liệu.
• Cho phép mỗi thành phần phát triển độc lập.
• Hỗ trợ đa nền tảng.
• Mỗi yêu cầu từ client phải có đủ thông tin cần thiết để server có thể hiểu được mà không cần phải lưu trữ thêm thông tin nào trước đó.
• Tất cả tài nguyên được truy cập thông qua một interface thống nhất (HTTP GET, PUT, POST, DELETE, ...).

2.5.4. Các dạng tương tác giữa Web Service với ứng dụng trên thiết bị di động

XML - eXtensible Markup Language

• Là ngôn ngữ đánh dấu với mục đích chung do W3C đề nghị.

• Là một dạng chuẩn cho phép lưu các thông tin hướng cấu trúc, được tổ chức dưới dạng thẻ (tag) tương ứng.
• Các thẻ (tag) của XML thường không được định nghĩa trước mà chúng được tạo ra theo quy ước của người, (hoặc Chương trình) tạo ra XML theo những quy ước của chính người tạo.
• Giúp đơn giản hóa việc chia sẻ dữ liệu giữa các hệ thống khác nhau, đặc biệt là các hệ thống được kết nối với Internet.
• Sử dụng các khai báo kiểu dữ liệu DTD (Document Type Definition) hay lược đồ Schema để mô tả dữ liệu.

Định dạng JSON - JavaScript Object Notation [4]

• Định nghĩa dữ liệu theo ngôn ngữ JavaScript, tiêu chuẩn ECMA-262 năm 1999.
• Là một định dạng văn bản đơn giản với các trường dữ liệu được lồng vào nhau.
• Dùng để trao đổi dữ liệu giữa các thành phần của một hệ thống tương thích với hầu hết các ngôn ngữ C, C++, C#, Java, JavaScript, Perl, Python...

Vì sao nên sử dụng JSON?

• Có thể đọc hiểu và dễ dàng tiếp cận (human-readability).
• Dữ liệu truyền tải ngắn gọn so với những định dạng dữ liệu khác như:

XML, HTML, … → Tiết kiệm dung lượng hơn XML, HTML,...

• Dễ dàng chuyển đổi (parse) dữ liệu từ dạng chuỗi (nhận từ server) sang dữ liệu có thể sử dụng được (thành Object, Number, Array). Dễ truy cập nội dung.
• Với những ứng dụng AJAX lấy và xử lý dữ liệu từ 1 web service nào đó khác domain. Nếu nội dung trả về có dạng JSON thì javascript từ trang web của chúng ta có thể trực tiếp truy cập (dùng lệnh eval).

CHƯƠNG 3: CHƯƠNG TRÌNH THỰC NGHIỆM

3.1. Giới thiệu

Các thiết bị di động ngày càng trở lên phổ biến đã đem lại nhiều lợi ích cho người dùng. Thay vì phải đến trực tiếp, người dùng chỉ cần thực hiện các kết nối và sử dụng các dịch vụ trên đó. Việc này giúp giảm thời gian, công sức và các chi phí không cần thiết khác. Mặt khác, hầu hết mối cá nhân hiện nay đều sở hữu một điện thoại thông minh, họ không chỉ dùng cho việc kết nối mà còn sử dụng các ứng dụng trên đó cho các công việc cá nhân của mình. Do đó, các phần mềm hiện nay được viết không chỉ sử dụng trên máy tính mà còn dùng cho các thiết bị di động đã mang lại nhiều lợi ích thiết thực cho người dùng trên nhiều lĩnh vực trong cuộc sống và lợi nhuận cho các công ty phát triển phần mềm.

Nhà ăn của Trường Đại học Quản lý và công nghệ Hải Phòng, phục vụ các suất ăn trưa cho cả cán bộ, giảng viên, nhân viên và sinh viên (gọi tắt là khách hàng) trong trường. Thông thường, mọi người cứ đến căng tin là được phục vụ. Các sinh viên sẽ trả tiền cho từng suất ăn của mình còn các cán bộ, giảng viên và nhân viên sẽ ký vào sổ để xác nhận đã ăn trưa để nhà trường thanh toán. Tuy nhiên sẽ xảy ra tình trạng có ngày sẽ thiếu món ăn để bán cho khách nhưng cũng có ngày lại thừa vì số người ăn có thể tăng giảm thất thường mà không được báo trước. Việc này ảnh hưởng trực tiếp đến lợi nhuận và khả năng phục vụ của căng tin và cũng gây bất tiện cho các khách hàng. Do đó, viêc xây dựng ứng dụng đăng ký ăn trưa cho cán bộ, giảng viên, nhân viên và sinh viên tại căng tin sẽ giúp việc theo dõi được số lượng khách ăn của của căng tin và cân đối lượng thực phẩm cần chuẩn bị trong ngày để việc cung cấp dịch vụ ngày càng tốt hơn và quản lý việc thu chi hiệu quả hơn. Người quản lý căng tin sẽ biết được có bao nhiêu người đăng ký ăn trong ngày hôm đó và dự kiên chuẩn bị thực phẩm phù hợp. Các khách hàng đăng ký ăn trước sẽ yên tâm khi luôn được phục vụ mà không cần phải đến tận nơi hay gọi điện để đăng ký.

3.2. Phát biểu bài toán

Nhà ăn của Trường ĐH QL và CN HP, phục vụ các suất ăn trưa cho cả cán bộ, giảng viên, nhân viên và sinh viên (khách hàng) trong trường.

Vào giờ ăn trưa, Khách hàng đến nhà ăn là được phục vụ. Tuy nhiên nếu quá đông khách hàng thì những người đến sau sẽ hết suất ăn, còn khi khách hàng ít lại thừa suất ăn. Việc này ảnh hưởng đến lợi nhuận và khả năng phục vụ của nhà ăn và cũng gây bất tiện cho khách hàng.

Xây dựng ứng dụng đăng ký ăn trưa tại trường Đại học Quản lý và Công nghệ Hải Phòng là cho phép khách hàng đăng ký ăn trưa mỗi ngày, và người quản lý biết được số lượng khách đặt ăn của ngày hôm đó để chuẩn bị và danh sách người đăng ký để đối chiếu với lượng cán bộ, giảng viên, nhân viên thực dự đến ăn và ký vào danh sách ăn trưa hay lượng sinh viên thực ăn ngày hôm đó.

Quy định thời gian đăng ký ăn trưa là trước 10 giờ sáng mỗi ngày. Nếu sau thời gian đó hệ thống từ chối và thông báo hết giờ đăng ký.

Các bước thực hiện trông hệ thống:

Cho phép người dùng đăng ký để đăng nhập hệ thống

Phân quyền cho người dùng gồm: quyền khách hàng dùng để đăng ký ăn trưa và quyền Admin cho người quản lý xem kết quả đăng ký và các thống kê báo cáo.

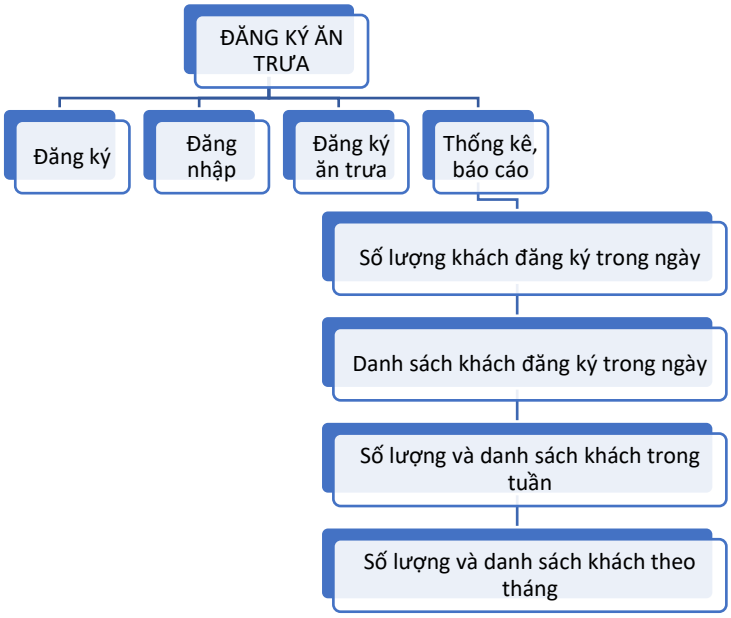
Cho khách hàng đăng ký ăn trưa trên ứng dụng

Cho người quản lý xem số lượng khách , thông tin khách , có thể xem thống kê trên ứng dụng danh sách và số lượng khách theo ngày ,theo khoảng thời gian , theo từng người…

3.3. Phân tích hệ thống

3.3.1 Ngữ cảnh hệ thống

3.3.2 Mô hình chức năng



Hình 3.2.1. Mô hình chức năng

Mô tả các chức năng:

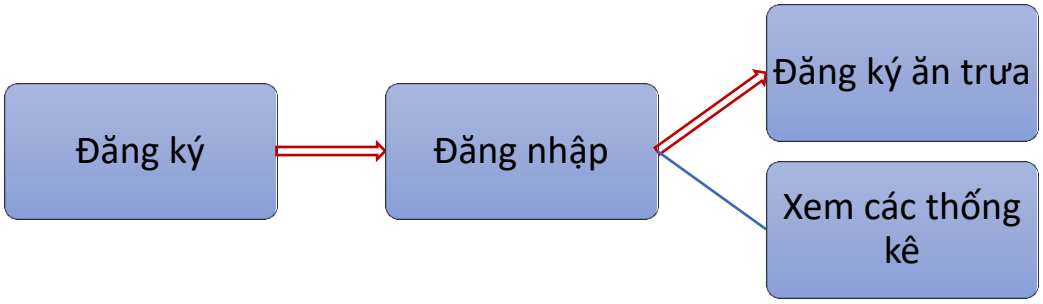
Đăng ký: Chức năng này dành cho các cán bộ, giảng viên, nhân viên và sinh viên trong trường đăng ký thông tin để đăng nhập vào hệ thống để đăng ký ăn trưa. Thông tin đăng ký bao gồm: Tên đăng nhập, mật khẩu, mã số, giới tính.

Đăng nhập: Để thực hiện đăng ký ăn trưa, các cán bộ, giảng viên, nhân viên và sinh viên phải đăng nhập vào hệ thống với tên đăng nhập và mật khẩu đã đăng ký.

Đăng ký ăn trưa: Sau khi cán bộ, giảng viên, nhân viên và sinh viên đăng nhập vào hệ thống, ứng dụng sẽ cho phép họ đăng ký ăn trưa. Nếu việc đăng ký thực hiện trước 10 giờ sáng ngày hôm đó, hệ thống sẽ thống báo cho cán bộ, giảng viên, nhân viên và sinh viên đã đăng ký thành công, ngược lại thì thông tin đăng ký ăn trưa của họ không được xác nhận.

Admin kiểm tra lượng khách: Ứng dụng sẽ hiển thị danh sách các giảng viên/ sinh viên đã đăng ký ăn trưa trong ngày hôm đó và thống kê tổng số người đăng ký.

3.3.3. Mô hình hoạt động



3.3.4. Cơ sở dữ liệu

3.3.4.1 Bảng thongtinkhach

Thiết kế cơ sở dữ liệu

Hình 3.2.1. Mô hình hoạt động

STT	Tên trường	Kiểu dữ liệu	Độ rộng	Mô tả	Ghi chú
1	iddangnhap	char	20	Định danh	Khóa chính
2	matkhau	char	20	Mật khẩu	
3	maso	char	11	Mã số	
4	hoten	char	30	Họ tên	
5	gioitinh	char	5	Giới tính	

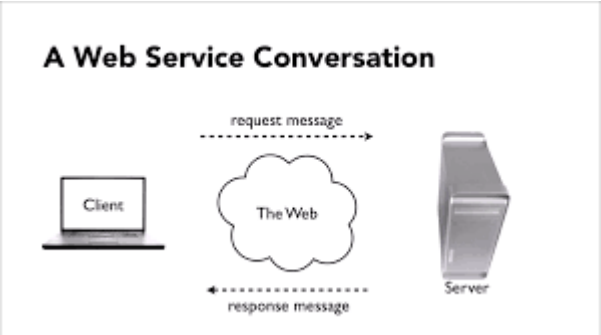
3.3.4.2 Bảng danhsachdangky

Thiết kế cơ sở dữ liêu

STT	Tên trường	Kiểu dữ liệu	Độ rộng	Mô tả	Ghi chú
2	iddangnhap	char	20	ID đăng nhập	Khóa chính
4	ngaydangky	Date time		Ngày đăng ký	Khóa chính

3.3.4.3 Đưa thông tin lên host

Sau khi đã bóc tách hay phân tích các thông tin về khách hàng, ta sẽ tiến hành đưa tất cả các thông tin trên lên Host. Và để có thể đưa được các thông tin trên lên Host, ta phải thông qua một bước trung gian là Webservice được viết bằng ngôn ngữ PHP



Hình 3.3.4.3.1 Mô hình Webservice

Trình tự các bước như sau :

a. Đưa các thông tin người dùng lên Host
b. Tiến hành xây dựng hàm xử lý đăng ký, đầu tiên là đẩy dữ liệu từ ứng dụng lên Webservice
c. Các hàm : đăng nhập , đăng ký ăn trưa , thống kê … làm tương tự
d. Chuyển thành định dạng Json (đã mã hóa thông tin)

```
protected Map<String, String> getParams() throws AuthFailureError {
    JSONArray jsonArray = new JSONArray();
    JSONObject jsonObject= new JSONObject();
    try {
        jsonObject.put("iddn", iddn);
        jsonObject.put("matkhau", matkhau);
        jsonObject.put("maso", maso);
        jsonObject.put("hoten", hoten);

        jsonObject.put("gioitinh", gioitinh);


    } catch (JSONException e) {
        e.printStackTrace();
    }
    jsonArray.put(jsonObject);


    HashMap<String, String> hashMap = new HashMap<>();
    hashMap.put("dangky_json", jsonArray.toString());
    return hashMap;
  }
};
RequestQueue requestQueue =
Volley.newRequestQueue(getApplicationContext());
requestQueue.add(stringRequest);
```

e.Tiến hành viết Webservice Đăng ký lên Host

```php
<?php
    include "connection.php";

    $DEBUGER = 0;

    if($DEBUGER == 0){
        $tinnhan_json = $_POST['dangky_json'];
        $tinnhan_info = json_decode($tinnhan_json, true);

        foreach ($tinnhan_info as $value) {
            $tendn = $value['tendn'];
            $matkhau = $value['matkhau'];
            $maso = $value['maso'];
            $hoten = $value['hoten'];
            $gioitinh = $value['gioitinh'];
        }

    }


    $query = "INSERT INTO dangky_json(tendn,matkhau,maso,hoten,gioitinh) VALUES ('$tendn', '$matkhau', '$maso', '$hoten',' $gioitinh')";

    // Thuc thi cau truy van them tin nhan moi
    $dangky_index = mysqli_query($link, $query);
    if($dangky_index){
        echo "1";
    } else {
        echo "0";
    }
?>
```

Các thông tin đăng ký gửi lên Host sẽ ở dạng file Json.

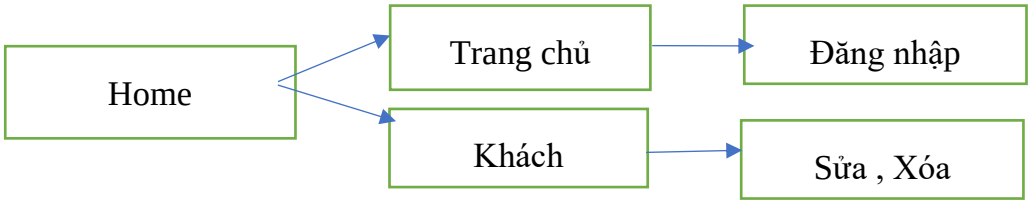
f. Lấy thông tin đăng ký từ Host về thông qua Webservice

```php
<?php
    include "connection.php";
    $listtinnhan=mysqli_query($link,'SELECT * FROM thongtinkhach');
    //if($listtinnhan-> num_rows > 0)
    {
        $arraythongtin=array();
        while($row=mysqli_fetch_array($listdangky))
        {
            $id = $row['id'];
            $tendn = $row['tendn'];
            $matkhau = $row['matkhau'];
            $maso  = $row['maso'];
            $hoten = $row['hoten'];
            $gioitinh = $value['gioitinh'];
            array_push($arraythongtin,new thongtinkhach($id,$tendn,$matkhau,$maso,$hoten,$gioitinh));
        }
        echo json_encode($arraythongtin);
    }
    class thongtinkhach
    {
        var $id;
        var $tendn;
        var $matkhau;
        var $maso,;
        var $hoten
        var $gioitinh
        function thongtinkhach($id,$tendn,$matkhau,$maso,$hoten,$gioitinh)
        {
            $this->id = $id;
            $this->tendn = $tendn;
            $this->matkhau = $matkhau;
            $this->maso = $maso;
            $this->hoten = $hoten;
            $this->gioitinh = $gioitinh;
        }
```

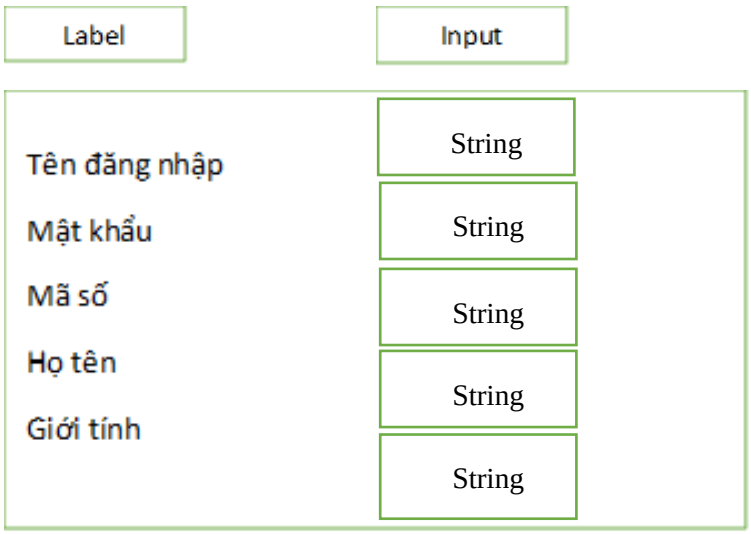
3.5. Thiết kế chương trình

3.5.1. Giao diện

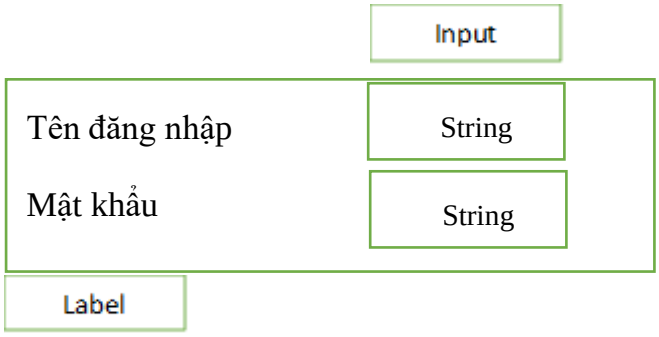
3.5.1.1 Giao diện chính



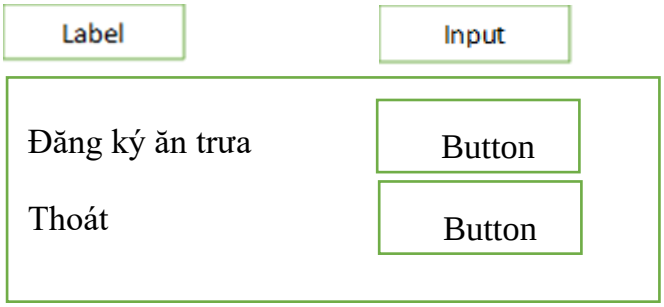
3.5.1.2 Giao diện đăng ký



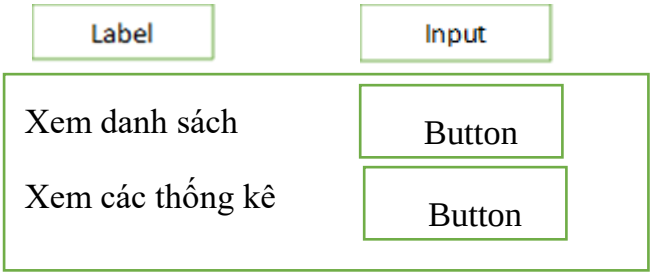
3.5.1.3 Giao diện đăng nhập



3.5.1.4 Giao diện đăng ký ăn trưa



3.5.1.5 Giao diện người quản lý



3.5.2 . Các chức năng chương trình

3.5.2.1 Chương trình cho phép người dùng :

- Đăng ký thông tin
- Truy nhập hệ thống
- Đăng ký ăn trưa
- Sửa thông tin đăng ký

3.5.2.1 Với người quản lý

Sửa thông tin đăng nhập

Xem danh sách ăn trưa

Xem thống kê về : - Lượng khách đăng ký ăn trưa theo ngày, theo từng người và theo khoảng thời gian.

- Thông tin khách hàng

Xóa tài khoản người dùng

3.6. Chương trình thực nghiệm

3.6.1 Thiết bị và Môi trường lập trình:

- Thiết bị: LapTop HP

  - Mainboard : Hewlett Packart - 1993
  - CPU : Core I5- 4300M
  - Ram :DDR 3, 8 GB
  - Ổ cứng : HDD 300 GB…

  Điện thoại SamSung J5 prime

- Môi trường lập trình: Hệ điều hành Windows 10

  Android Studio 4.0, SDK 8.1

  Ngôn ngữ lập trình Java

3.6.2. Ứng dụng đăng ký ăn trưa

Một số giao diện và hoạt động chính của chương trình

3.6.2.1 Đăng ký tài khoản:

Đăng ký thông tin bao gồm:

- ID đăng nhập
- Mật khẩu
- Mã sinh viên / cán bộ / Giảng viên …
- Họ tên
- Giới tính

3.6.2.2 Đăng nhập hệ thống

Sau khi đăng ký thành công ta có thể đăng nhập vào hệ thống

Thông tin đăng nhập bao gồm

- Tên đăng nhập
- Mật khẩu

3.6.2.3 Đăng ký ăn trưa

Do việc chuẩn bị bữa ăn cần nhiều thời gian nên việc đăng ký ăn trưa chỉ được thực hiện trong khoảng thời gian cho phép của ứng dụng.

Khách hàng chỉ có thể đăng ký trong khoảng 7 giờ đến 10 giờ hằng ngày

Ngoài thời gian trên ứng dụng sẽ không cho đăng ký ăn trưa

Trong khoảng thời gian 7 giờ đến 10h

Khách hàng có thể đăng ký ăn trưa

Khách hàng tiến hành Click vào nút **ĐĂNG KÝ ĂN TRƯA**

Ngoài khoảng thời gian trên

Ứng dụng sẽ khóa chức năng đăng ký ăn trưa

3.6.2.4 Thống kê, báo cáo

Người quản lý đăng nhập dưới tài khoản Admin và click vào nút **THỐNG KÊ** để xem thống kê theo tùy chọn :

Sau khi Click vào nút thống kê ứng dụng sẽ đưa vào giao diện Thống Kê như :

* Để xem được các thống kê phía trên thì các thông tin khách hàng cần được lưu trữ lại trên bảng **danhsachdangky** đã được tạo trong cơ sở dữ liệu bao gồm các thông tin :

- Tên đăng nhập
- Họ tên
- Ngày đăng ký

Khi khách hàng Click vào nút **Đăng ký ăn trưa** hệ thống sẽ lưu những thông tin phía trên vào bảng **danhsachdangky** từ đó muốn xem thống kê gì ta chỉ cần sử dụng các lệnh truy vấn cơ sở dữ liêu có dạng :

SELECT< cột…> FROM < tên bảng > WHERE < điều kiện...>

3.7 Các yêu cầu đối với người dùng hệ thống.

- Người sử dụng phải là cán bộ, giảng viên, nhân viên và sinh viên trong trường
- Người dùng phải có thiết bị di động sử dụng hệ điều hành Android, ví dụ như :
  - Điện thoại : Sam sung , Oppo , Xiaomi , Vinsmart , Nexus,…
  - Máy tính bảng : Samsung Galaxy Tab , Masstel Tab , Huawei MediaPad , Lenovo Yoga Tab …
  - Và một số thiết bị khác

KẾT LUẬN

1. Kết quả đạt được của đồ án :

- Trong thời gian thực hiện đề tài, em đã tìm hiểu cài đặt và học lập trình Android để có thể viết được một ứng dụng đăng ký ăn trưa tại Trường Đại học Quản lý và Công nghệ Hải Phòng qua thiết bị di động.
- Chương trình đã thực hiện được các chức năng cơ bản như: cán bộ, giảng viên, nhân viên và sinh viên trong trường đăng ký thông tin, đăng ký thông tin để đăng nhập vào hệ thống; Phân quyền sử dụng: cán bộ, giảng viên, nhân viên và sinh viên đăng nhập để đăng ký ăn trưa, người quản lý đăng nhập để lấy số liệu báo cáo về số lượng người và danh sách những người đã đăng ký trong ngày; Hệ thống chỉ cho phép việc đăng ký trước 10 giờ hàng ngày để người quản lý nắm được số lượng khách hàng và chuẩn bị thực phẩm trong ngày hôm đó.
- Thông qua kết quả của việc đăng ký ăn trưa, nhà ăn nhanh chóng nắm được số lượng người ăn trưa để chủ động trong việc cung cấp bữa ăn. Việc này giúp khả năng phục vụ của nhà ăn tốt hơn và cán bộ, giảng viên, nhân viên và sinh viên cũng đảm bảo chắc chắn mình được phục vụ khi đã đăng ký trực tuyến mà không cần đến tận nơi hay gọi điện để thông báo.

2. Những hạn chế :

- Trong khoảng thời gian ngắn để thực đề tài, em mới chỉ xây dựng được ứng dụng với các chức năng cơ bản theo yêu cầu của hệ thống.
- Ứng dụng hạn chế về giao diện và các thông điệp trao đổi khi người dùng tương tác

3. Hướng phát triển tiếp theo:

Trong thời gian tới em sẽ chỉnh sửa lại giao diện cho gần gũi dễ sử dụng, và thiết kế các thống kê báo cáo bổ sung