BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGHÀNH CÔNG NGHỆ THÔNG TIN

HẢI PHÒNG 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

XÂY DỰNG ỨNG DỤNG QUẢN LÝ BÀN ĂN TRÊN THIẾT BỊ DI ĐỘNG ANDROID

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ Thông tin

HẢI PHÒNG 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

XÂY DỰNG ỨNG DỤNG QUẢN LÝ BÀN ĂN TRÊN THIẾT BỊ DI ĐỘNG ANDROID

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ Thông tin

Sinh viên thực hiện: Khoa Tùng Dương

Giáo viên hướng dẫn: Th.S Phùng Anh Tuấn

Mã sinh viên: 1412102046

HẢI PHÒNG 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

-------o0o------

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

Sinh viên: Khoa Tùng Dương Mã số: 1412102046

Lớp: CT1901C Ngành: Công nghệ Thông tin

Tên đề tài: Xây dựng ứng dụng quản lý bàn ăn trên thiết bị di động Android

NHIỆM VỤ ĐỀ TÀI

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

a. Nội dung:

- Tìm hiểu hệ điều hành android.

- Tìm hiểu môi trường lập trình Android Studio.

- Tìm hiểu về quy trình nghiệp vụ quản lý bàn ăn tại nhà hàng.

- Ứng dụng các kỹ thuật đã tìm hiểu xây dựng chương trình ứng dụng thực nghiệm.

b. Các yêu cầu cần giải quyết

- Sử dụng công cụ Android Studio để viết chương trình chạy trên điện thoại android.

- Xây dựng được chương trình thực nghiệm quản lý bàn ăn tại nhà hàng, đóng gói chương trình thành file *.apk cho phép cài đặt và chạy trên điện thoại android thật.

2. Các số liệu cần thiết để thiết kế, tính toán

- Sử dụng các số liệu giả lập.

3. Địa điểm thực tập

Trường Đại học Dân lập Hải Phòng

CÁN BỘ HƯỚNG DẪN ĐỀ TÀI TỐT NGHIỆP

Người hướng dẫn thứ nhất:

Họ và tên: Phùng Anh Tuấn

Học hàm, học vị: Thạc sỹ

Cơ quan công tác: Trường Đại học Dân lập Hải Phòng

Nội dung hướng dẫn:

- Tìm hiểu hệ điều hành android.
- Tìm hiểu môi trường lập trình Android Studio.
- Tìm hiểu về quy trình nghiệp vụ quản lý bàn ăn tại nhà hàng.
- Ứng dụng các kỹ thuật đã tìm hiểu xây dựng chương trình ứng dụng thực nghiệm

Người hướng dẫn thứ hai:

Họ và tên:

Học hàm, học vị:

Cơ quan công tác:

Nội dụng hướng dẫn:

Đề tài tốt nghiệp được giao ngày 01 tháng 07 năm 2019

Yêu cầu phải hoàn thành trước ngày 21 tháng 09 năm 2019

Đã nhận nhiệm vụ: Đ.T.T.N	Đã nhận nhiệm vụ: Đ.T.T.N
Sinh viên	Cán bộ hướng dẫn Đ.T.T.N

Hải phòng, ngày .....tháng.....năm 2019

HIỆU TRƯỞNG

GS.TS.NGƯT Trần Hữu Nghị

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: Phùng Anh Tuấn.

Đơn vị công tác: Trường Đại học Dân lập Hải Phòng.

Họ và tên sinh viên: Khoa Tùng Dương - Ngành: Công nghệ Thông tin

Nội dung hướng dẫn:

- Tìm hiểu hệ điều hành android.
- Tìm hiểu môi trường lập trình Android.
- Tìm hiểu về quy trình nghiệp vụ quản lý bàn ăn tại nhà hàng.
- Ứng dụng các kỹ thuật đã tìm hiểu xây dựng chương trình ứng dụng thực nghiệm

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

- Có nghiên cứu tài liệu phục vụ cho nội dung đồ án.
- Khả năng làm việc độc lập còn nhiều hạn chế.
- Chưa làm chủ được nội dung đồ án.

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...)

- Đã thực hiện được các nội dung đề ra trong đề cương.
- Nội dung đồ án có tính thực tế.
- Cần trình bày nội dung đồ án rõ ràng, logic hơn.

3. Ý kiến của giảng viên hướng dẫn tốt nghiệp

Đạt ☐ Không đạt ☐ Điểm:……...

Hải Phòng, ngày 30 tháng 09 năm 2019

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

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 2019

Giảng viên chấm phản biện

(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Lời đầu tiên em xin chân thành cảm ơn các thầy, cô trong khoa Công nghệ thông tin, trường Đại học Dân lập Hải Phòng đã tạo điều kiện thuận lợi cho em trong quá trình học tập tại trường cũng như trong thời gian thực hiện đồ án tốt nghiệp. Đặc biệt, em muốn gửi lời cảm ơn tới Thạc Sĩ – Phùng Anh Tuấn giảng viên trực tiếp hướng dẫn, chỉ bảo giúp em khắc phục những khó khăn, thiếu sót để có thể hoàn thành các phần trong đồ án tốt nghiệp từ lý thuyết cho tới thực hành sử dụng công cụ.

Mặc dù đã cố gắng với tất cả nỗ lực của bản thân để hoàn thiện đồ án, nhưng do thời gian có hạn, năng lực và kinh nghiệm còn hạn chế nên đồ án không thể tránh khỏi những thiếu sót. Kính mong nhận được sự đóng góp ý kiến từ phía thầy cô, bạn bè để em có thể nâng cao kiến thức của bản thân, hoàn thiện đồ án được tốt hơn.

Em xin chân thành cảm ơn!

Hải Phòng, ngày 21 tháng 09 năm 2019.

Sinh viên thực hiện

Khoa Tùng Dương

MỤC LỤC

LỜI CẢM ƠN

MỤC LỤC

MỞ ĐẦU 1

DANH MỤC HÌNH VẼ 2

CHƯƠNG 1 GIỚI THIỆU VỀ HỆ ĐIỀU HÀNH ANDROID 4

1.1. Lịch sử ra đời.....4

1.2. Giao diện 5

1.3. Ứng dụng.....6

1.4. Kiến trúc hệ điều hành Android 7

1.4.1. Tầng hạt nhân Linux 7

1.4.2. Tầng thư viện 8

1.4.3. Tầng thực thi các ứng dụng trên hệ điều hành Android.....8

1.4.4. Tầng khung ứng dụng 9

1.4.5. Tầng ứng dụng 10

1.5. Quản lý bộ nhớ 10

1.6. Cập nhật.....11

1.7. Bảo mật và riêng tư 11

1.8. Ưu và nhược điểm 12

1.8.1. Ưu điểm.....12

1.8.2. Nhược điểm.....12

1.9. Thiết bị.....12

1.10. Kết luận 13

CHƯƠNG 2 MÔI TRƯỜNG LẬP TRÌNH ANDROID STUDIO 14

2.1. Thiết lập môi trường.....14

2.1.1. Cài đặt Java 14
2.1.2. Cài đặt Android Studio.....15
2.2. Các thành phần trong một dự án Android 15
2.2.1. File manifest.....15
2.2.2. File java.....16
2.2.3. File tài nguyên.....17
2.2.4. File Grandle.....17
2.3. Chu kỳ ứng dụng Android.....18
2.3.1. Chu kỳ sống thành phần.....18
2.3.2. Ngăn xếp Activity 18
2.3.3. Vòng đời của Activity.....19
2.3.4. Các phương thức của chu kỳ sống 19
2.4. View.....21
2.4.1. TextView.....21
2.4.2. Button.....21
2.4.3. ImageButton.....22
2.4.4. ImageView 22
2.4.5. EditText.....22
2.4.6. ListView.....22
2.4.7. GridView.....24
2.5. ViewGroup 25
2.5.1. LinearLayout.....25
2.5.2. RelativeLayout.....25
2.5.3. FrameLayout 25
2.5.4. TableLayout 25

2.6. Activity & Intent....26
2.6.1. Activity....26
2.6.2. Intent....26
CHƯƠNG 3 KỸ THUẬT XÂY DỰNG ỨNG DỤNG....28
3.1. Hệ quản trị cơ sở dữ liệu SQLite....28
3.2. Các thao tác cơ bản trên cơ sở dữ liệu....28
3.2.1. Tạo cơ sở dữ liệu....28
3.2.2. Các thao tác trên cơ sở dữ liệu....29
3.3. Quản lý người dùng....32
3.3.1. Thực hiện đăng ký người dùng....32
3.3.2. Thay đổi thông tin người dùng....32
3.3.3. Xóa thông tin người dùng....33
3.3.4. Lấy danh sách người dùng....33
3.3.5. Kiểm tra đăng nhập....33
3.4. Quản lý bàn ăn....34
3.4.1. Thêm bàn ăn....34
3.4.2. Thay đổi tên bàn ăn....34
3.4.3. Xóa bàn ăn....34
3.4.4. Lấy danh sách bàn ăn....35
3.5. Quản lý thực đơn....35
3.5.1. Thêm loại món ăn....35
3.5.2. Lấy danh sách loại món ăn....36
3.5.3. Thêm món ăn....36
3.5.4. Lấy danh sách món ăn....37
3.6. Quản lý thống kê....37

3.6.1. Lấy danh sách thống kê theo ngày 37
3.6.2. Lấy danh sách thống kê từ ngày đến ngày và thống kê tháng 38
CHƯƠNG 4 CHƯƠNG TRÌNH ỨNG DỤNG THỰC NGHIỆM 39
4.1. Phát biểu bài toán 39
4.2. Sơ đồ chức năng 39
4.3. Cơ sở dữ liệu.....40
4.4. Giao diện chương trình.....42
4.4.1. Giao diện đăng ký 42
4.4.2. Giao diện đăng nhập.....43
4.4.3. Giao diện Menu.....43
4.4.4. Giao diện bàn ăn.....44
4.4.5. Giao diện thêm bàn ăn.....44
4.4.6. Giao diện sửa tên bàn ăn 45
4.4.7. Giao diện xóa bàn ăn.....46
4.4.8. Giao diện loại món ăn 47
4.4.9. Giao diện món ăn 47
4.4.10. Giao diện thêm món ăn 48
4.4.11. Giao diện thêm loại món ăn 48
4.4.12. Giao diện chọn ảnh món ăn.....49
4.4.13. Giao diện danh sách nhân viên.....49
4.4.14. Giao diện cập nhật thông tin nhân viên.....50
4.4.15. Giao diện xóa nhân viên.....51
4.4.16. Giao diện gọi món.....51
4.4.17. Giao diện bàn ăn đổi màu sau khi gọi món ăn 53
4.4.18. Giao diện Thanh toán 54

4.4.19. Giao diện thống kê 55
4.4.20. Giao diện thống kê theo ngày.....55
4.4.21. Giao diện thống kê từ ngày đến ngày.....56
4.4.22. Giao diện thống kê theo tháng 56
4.4.23. Giao diện thoát ứng dụng.....57
KẾT LUẬN.....58
DANH MỤC TÀI LIỆU THAM KHẢO.....59

MỞ ĐẦU

Khi nền kinh tế phát triển, ngành công nghệ thông tin trở thành một trợ thủ, điều không thể thiếu với hầu hết các nghề. Có thể ban đầu, nhiều người sẽ cảm thấy chưa cần thiết vì chỉ kinh doanh nhỏ, hoặc chưa sẵn sàng để sử dụng vì nghĩ rằng phần mềm là một gì đó rất phức tạp, rất khó dùng.

Việc quản lý theo dõi bàn ăn tại nhà hàng bằng hóa đơn theo cách truyền thống vẫn tồn tại nhiều điểm bất cập, một trong những bất cập được quan tâm là khâu tính tiền thanh toán bàn ăn cho khách cần phải nhanh chóng và chính xác.

Ứng dụng di động quản lý bàn ăn trên nền hệ điều hành Android hỗ trợ các công việc kinh doanh cho nhà hàng. Giúp nhân viên phục vụ có thể đặt món ăn, tránh sai sót trong việc thanh toán, kiểm soát hoạt động doanh thu ngay trên chiếc điện thoại, máy tính bảng.

Nội dung đề tài bao gồm 5 phần như sau:

- Chương 1: Giới thiệu về hệ điều hành Android.

- Chương 2: Môi trường lập trình Android Studio.

- Chương 3: Kỹ thuật xây dựng ứng dụng.

- Chương 4: Chương trình ứng dụng thực nghiệm.

- Kết luận: Phần này đưa ra những kết quả đồ án đạt được, những thiếu sót chưa thực hiện được và hướng phát triển đề tài trong tương lai.

DANH MỤC HÌNH VẼ

Hình 1.2.1: Thanh trạng thái

Hình 1.3.1: Kho ứng dụng

Hình 1.4.1: Kiến trúc hệ điều hành Android

Hình 2.1.1.1: Cài đặt JDK

Hình 2.1.2.1: Cài đặt Android Studio

Hình 2.3.2.1: Ngăn xếp Activity

Hình 2.3.3.1: Vòng đời của Activity

Hình 2.4.6.1: Hình minh họa ListView

Hình 2.4.7.1: Hình minh họa GridView

Hình 2.6.2.1: Truyền dữ liệu giữa 2 Activity

Hình 4.2.1: Sơ đồ chức năng

Hình 4.4.1.1: Giao diện đăng ký

Hình 4.4.2.1: Màn hình đăng nhập

Hình 4.4.3.1: Giao diện Menu

Hình 4.4.4.1: Giao diện bàn ăn

Hình 4.4.5.1: Giao diện thêm bàn ăn

Hình 4.4.6.1: Giao diện sửa tên bàn ăn

Hình 4.4.7.1: Giao diện xóa bàn ăn

Hình 4.4.8.1: Giao diện loại món ăn

Hình 4.4.9.1: Giao diện món ăn

Hình 4.4.10.1: Giao diện thêm thực đơn

Hình 4.4.11.1: Giao diện thêm loại món ăn

Hình 4.4.12.1: Chọn ảnh món ăn

Hình 4.4.13.1: Giao diện nhân viên

Hình 4.4.14.1: Giao diện cập nhật thông tin nhân viên

Hình 4.4.15.1: Giao diện xóa nhân viên

Hình 4.4.16.1: Giao diện gọi món

Hình 4.4.17.1: Giao diện gọi số lượng món

Hình 4.4.18.1: Giao diện thanh toán

Hình 4.4.19.1: Giao diện thống kê

Hình 4.4.20.1: Giao diện thống kê theo ngày

Hình 4.4.21.1: Giao diện Thống kê từ ngày đến ngày

Hình 4.4.22.1: Giao diện thống kê tháng

Hình 4.4.23.1: Giao diện thoát ứng dụng

CHƯƠNG 1
GIỚI THIỆU VỀ HỆ ĐIỀU HÀNH ANDROID

1.1. Lịch sử ra đời

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 Android, Inc. Với sự hỗ trợ tài chính của Google và sau này được chính Google mua lại vào năm 2005 và hệ điều hành Android đã ra mắt vào năm 2007. Chiếc điện thoại đầu tiên chạy Android là HTC Dream được bán vào ngày 22 tháng 10 năm 2008. Biểu trưng của hệ điều hành Android mới là một con robot màu xanh lá cây do hãng thiết kế Irina Blok tại California vẽ.

Chính mã nguồn mở cùng với giấy phép không có nhiều ràng buộc đã cho phép các nhà phát triển thiết bị, mạng di động và các lập trình viên được điều chỉnh và phân phối Android một cách tự do. Những yếu tố này đã giúp Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới. Android chiếm 75% thị phần điện thoại thông minh trên toàn thế giới vào thời điểm quý 3 năm 2012, với tổng cộng 500 triệu thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày. Tháng 10 năm 2012, đã có khoảng 700.000 ứng dụng trên Android, và số lượt tải ứng dụng từ Google Play (cửa hàng ứng dụng chính của Android) ước tính khoảng 25 tỉ lượt. Hiện nay con số này đã giảm xuống do sự ảnh hướng lớn của iOS từ Apple và một phần nhỏ của Windows Phone, tuy nhiên Android vẫn dẫn đầu thị phần.

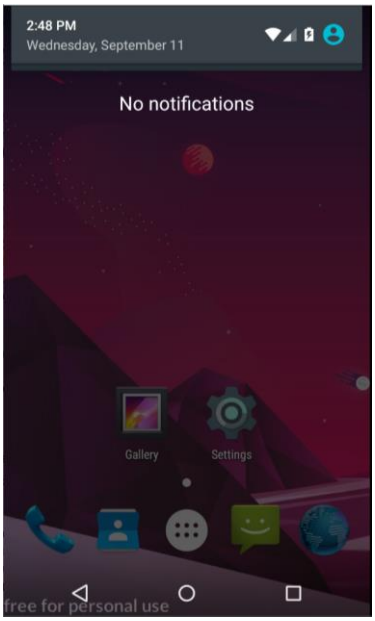
Từ năm 2008, Android đã trải qua nhiều lần cập nhật để dần dần cải tiến hệ điều hành, bổ sung các tính năng mới và sửa các lỗi trong những lần phát hành trước. Mỗi bản nâng cấp được đặt tên lần lượt theo thứ tự bảng chữ cái, theo tên của một món ăn tráng miệng; ví dụ như phiên bản 1.5 Cupcake (bánh bông lan nhỏ có kem) tiếp nối bằng phiên bản 1.6. Phiên bản mới nhất hiện nay là 9.0 Pie ra mắt vào tháng 8 năm 2018. Vào năm 2010, Google ra mắt loạt thiết bị Nexus một dòng sản phẩm bao gồm điện thoại thông minh và máy tính bảng chạy hệ điều hành Android, do các đối tác phần cứng sản xuất. HTC đã hợp tác với Google trong chiếc điện thoại thông minh Nexus đầu tiên, Nexus One. Kể từ đó nhiều thiết bị mới hơn đã gia nhập vào dòng sản phẩm này, như điện thoại Nexus 4 và máy tính bảng Nexus 10, lần lượt do LG và Samsung sản xuất. Google xem điện thoại và máy tính bảng Nexus là những thiết bị Android chủ lực của mình, với những tính năng phần cứng và phần mềm mới nhất của Android.[1]

1.2. Giao diện

Giao diện người dùng của Android dựa trên nguyên tắc tác động trực tiếp, sử dụng cảm ứng chạm tương tự như những động tác ngoài đời thực như vuốt, chạm, kéo giãn và thu lại để xử lý các đối tượng trên màn hình.

Các thiết bị Android sau khi khởi động sẽ hiển thị màn hình chính, điểm khởi đầu với các thông tin chính trên thiết bị, tương tự như khái niệm desktop trên máy tính để bàn. Màn hình chính Android thường gồm nhiều biểu tượng(icon) và tiện ích(widget). Giao diện màn hình chính của Android có thể tùy chỉnh ở mức cao, cho phép người dùng tự do sắp đặt hình dáng cũng như hành vi của thiết bị theo sở thích.

Những ứng dụng do các hãng thứ ba có trên Google Play và các kho ứng dụng khác còn cho phép người dùng thay đổi "chủ đề" của màn hình chính, thậm chí bắt chước hình dáng của hệ điều hành khác như Windows Phone hay iOS. Phần lớn những nhà sản xuất và một số nhà mạng, thực hiện thay đổi hình dáng và hành vi của các thiết bị Android của họ để phân biệt với các hãng cạnh tranh.



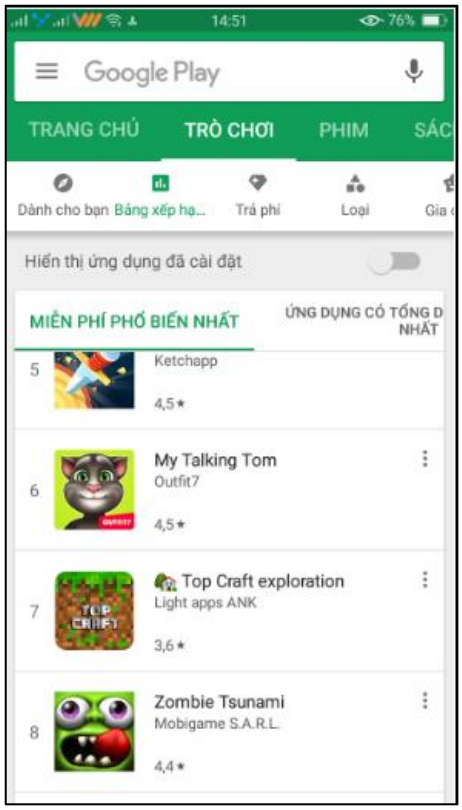
Hình 1.2.1: Thanh trạng thái

Ở phía trên cùng màn hình là thanh trạng thái, hiển thị thông tin về thiết bị và tình trạng kết nối. Thanh trạng thái này có thể "kéo" xuống để xem màn hình thông báo gồm thông tin quan trọng hoặc cập nhật của các ứng dụng, như email hay tin nhắn SMS mới nhận, mà không làm gián đoạn hoặc khiến người dùng cảm thấy bất tiện. Trong các phiên bản đời đầu, người dùng có thể nhấn vào thông báo để mở ra ứng

dụng tương ứng, về sau này các thông tin cập nhật được bổ sung thêm tính năng, như có khả năng lập tức gọi ngược lại khi có cuộc gọi nhỡ mà không cần mở ứng dụng gọi điện. Thông báo sẽ luôn nằm trong đó cho đến khi người dùng đã đọc hoặc xóa nó đi.[1]

1.3. Ứng dụng

Android có lượng ứng dụng của bên thứ ba ngày càng nhiều, được chọn lọc và đặt tên một cửa hàng ứng dụng như Google Play hay Amazon Appstore để người dùng lấy về, hoặc bằng cách tải xuống rồi cài đặt tập tin APK từ trang web khác. Các ứng dụng trên Play Store cho phép người dùng duyệt, tải về và cập nhật các ứng dụng do Google và các nhà phát triển thứ ba phát hành. Play Store được cài đặt sẵn trên các thiết bị thỏa mãn điều kiện tương thích của Google. Ứng dụng sẽ tự động lọc ra một danh sách các ứng dụng tương thích với thiết bị của người dùng, và nhà phát triển có thể giới hạn ứng dụng của họ chỉ dành cho những nhà mạng cố định hoặc những quốc gia cố định vì lý do kinh doanh.



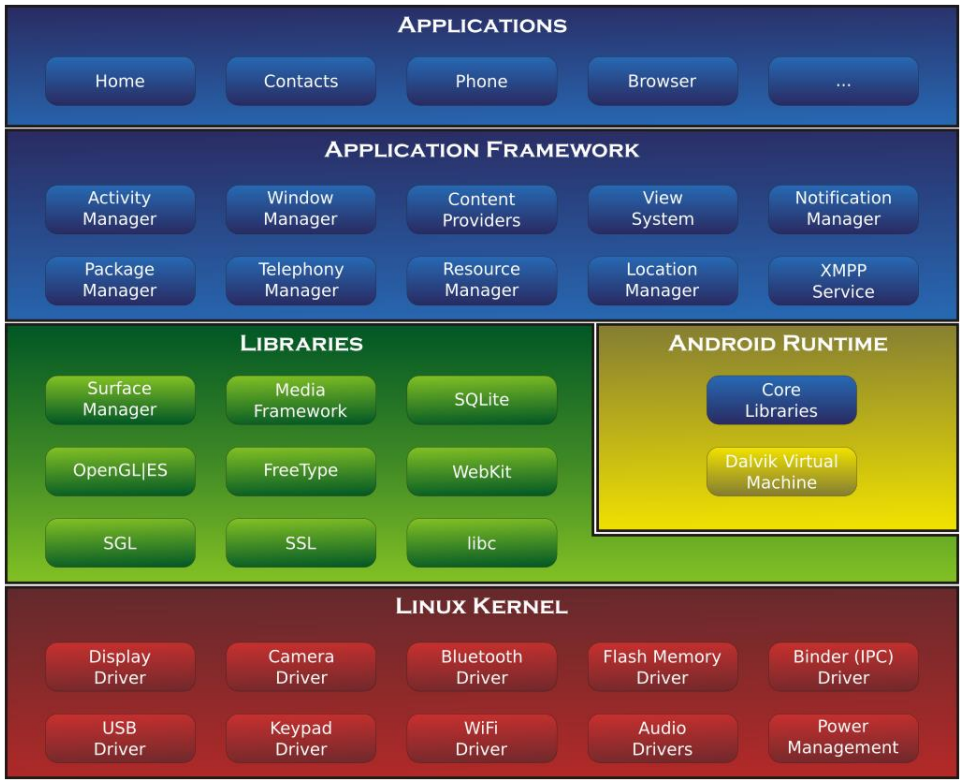
Hình 1.3.1: Kho ứng dụng

Các ứng dụng cho Android được phát triển bằng ngôn ngữ Java sử dụng bộ phát triển phần mềm Android (SDK). SDK bao gồm một bộ đầy đủ các công cụ dùng để phát triển, gồm có công cụ gỡ lỗi, thư viện phần mềm, bộ giả lập điện thoại dựa

trên QEMU, tài liệu hướng dẫn, mã nguồn mẫu, và hướng dẫn từng bước. Môi trường phát triển tích hợp được hỗ trợ chính thức là Android Studio, một môi trường đồ họa cho những nhà lập trình mới bắt đầu, và nhiều nền tảng ứng dụng di động đa nền tảng phong phú.[1]

1.4. Kiến trúc hệ điều hành Android

Hệ điều hành Android có 4 tầng từ dưới lên trên là tầng hạt nhân Linux (phiên bản 2.6), tầng thư viện & tầng thực thi các ứng dụng trên hệ điều hành Android, tầng khung ứng dụng và trên cùng là tầng ứng dụng.



Hình 1.4.1: Kiến trúc hệ điều hành Android

1.4.1. Tầng hạt nhân Linux

Hệ điều hành Android được phát triển dựa trên hạt nhân Linux, cụ thể là hạt nhân Linux phiên bản 2.6, điều đó được thể hiện ở lớp dưới cùng này. Tất cả mọi hoạt động của điện thoại muốn thi hành được thì đều được thực hiện ở mức cấp thấp ở lớp này bao gồm quản lý bộ nhớ (memory management), giao tiếp với phần cứng (driver model), thực hiện bảo mật (security), quản lý tiến trình (process).

Tuy được phát triển dựa vào nhân Linux nhưng thực ra nhân Linux đã được nâng cấp và sửa đổi rất nhiều đề phù hợp với tính chất của những thiết bị cầm tay như hạn chế về bộ vi xử lý, dung lượng bộ nhớ, kích thước màn hình, nhu cầu kết nối mạng không

dây…

Tầng này có các thành phần chủ yếu:

- Display Driver: Điều khiển việc hiển thị lên màn hình cũng như thu thập những điều khiển của người dùng lên màn hình (di chuyển, cảm ứng).

- Camera Driver: Điều khiển hoạt động camera, nhận luồng dữ liệu từ camera trả về.

- Bluetooth Driver: Điều khiển thiết bị phát và thu sóng Bluetooth.

- USB Driver: Quản lý hoạt động của các cổng giao tiếp USB.

- Keypad Driver: Điều khiển bàn phím.

- Wi-Fi Driver: Chịu trách nhiệm về việc thu phát sóng Wi-Fi.

- Audio Driver: Điều khiển các bộ thu phát âm thanh, giải mã các tín hiệu dạng audio thành tín hiệu số và ngược lại.

- Binder IPC Driver: Chịu trách nhiệm về việc kết nối và liên lạc với mạng vô tuyến như CDMA, GSM, 3G, 4G, E để đảm bảo những chức năng truyền thông được thực hiện.

- M-system Driver: Quản lý việc đọc ghi… lên các thiết bị nhớ như thẻ SD, flash.

- Power Management: Giám sát việc tiêu thụ điện năng.

1.4.2. Tầng thư viện

Phần này có nhiều thư viện được viết bằng C/C++ để các phần mềm có thể sử dụng, các thư viện đó được tập hợp thành một số nhóm như:

- Thư viện hệ thống (System C library): thư viện dựa trên chuẩn C, được sử dụng chỉ bởi hệ điều hành.

- Thư viện Media (Media Libraries): Có nhiều code để hỗ trợ việc phát và ghi các loại định dạng âm thanh, hình ảnh, video thông dụng.

- Thư viện web (LibWebCore): Đây là thành phần để xem nội dung trên web, được sử dụng để xây dựng phần mềm duyệt web (Android Browse) cũng như để các ứng dụng khác có thể nhúng vào. Nó cực kỳ mạnh, hỗ trợ được nhiều công nghệ mạnh mẽ như HTML5, JavaScript, CSS, DOM, AJAX.

- Thư viện SQLite: Hệ cơ sở dữ liệu để các ứng dụng có thể sử dụng.

1.4.3. Tầng thực thi các ứng dụng trên hệ điều hành Android

Phần này chứa các thư viện mà một chương trình viết bằng ngôn ngữ Java có thể hoạt động. Phần này có 2 bộ phận tương tự như mô hình chạy Java trên máy tính thường. Thứ nhất các thư viện lõi (Core Library), chứa các lớp như JAVA IO, collections, File

Access. Thứ hai là một máy ảo java (Dalvik Virtual Machine).

Mặc dù cũng được viết từ ngôn ngữ Java nhưng một ứng dụng Java của hệ điều hành Android không được chạy bằng JRE của Sun (nay là Oracle) (JVM) mà là chạy bằng máy ảo Dalvik do Google phát triển.

1.4.4. Tầng khung ứng dụng

Tầng này xây dựng bộ công cụ - các phần tử ở mức cao để các lập trình viên có thể nhanh chóng xây dựng ứng dụng. Nó được viết bằng Java, có khả năng sử dụng chung để tiết kiệm tài nguyên.

Đây là một nền tảng mở, điều đó có 2 điều lợi:

- Với các hãng sản xuất điện thoại: Có thể tùy biến để phù hợp với cấu hình điện thoại mà họ sản xuất cũng như để có nhiều mẫu mã, style hợp thị hiếu người dùng. Vì thế nên tuy cùng chung nền tảng android mà điện thoại của Google có thể khác hẳn với Motorola, HTC, T-Mobile, Samsung.

- Với lập trình viên: Cho phép lập trình viên có thể sử dụng các API ở tầng trên mà không cần phải hiểu rõ cấu trúc bên dưới, tạo điều kiện cho lập trình viên tự do sáng tạo bởi vì chỉ cần quan tâm đến nội dung mà ưng dụng họ làm việc. Một tập hợp API rất hữu ích được xây dựng sẵn như hệ thống định vị, các dịch vụ chạy nền, liên lạc giữa các ứng dụng, các thành phần giao diện cấp cao.

Giới thiệu một số thành phần của phần này:

- Activity Manager: Quản lý các chu kỳ sống của một ứng dụng cũng như cung cấp công cụ điều khiển các Activity.

- Telephony Manager: Cung cấp công cụ để thực hiện việc liên lạc như gọi điện thoại.

- XMPP Service: Cung cấp công cụ để liên lạc trong thời gian thực.

- Location Manager: Cho phép xác định vị trí của điện thoại dựa vào hệ thống định vị toàn cầu GPS và Google Maps.

- Window Manager: Quản lý việc xây dựng và hiển thị các giao diện người dùng cũng như tổ chức quản lý các giao diện giữa các ứng dụng.

- Notification Manager: Quả lý việc hiển thị các thông báo (như báo có tin nhắn, có e-mail mới).

- Resource Manager: Quản lý tài nguyên tĩnh của các ứng dụng bao gồm các file hình

ảnh, âm thanh, layout, string. (Những thành phần không được viết bởi ngôn ngữ lập trình).

1.4.5. Tầng ứng dụng

Đây là lớp ứng dụng giao tiếp với người dùng, bao gồm các ứng dụng như:

- Các ứng dụng cơ bản, được cài đặt đi liền với hệ điều hành là gọi điện (Phone), quản lý danh bạ(Contacts), duyệt web (Browser), nhắn tin (SMS), lịch làm việc (Calendar), đọc e-mail (Email-Client), bản đồ (Map), quay phim chụp ảnh (Camera)...

- Các ứng dụng được cài thêm như các phần mềm chứng khoán (Stock), các trò chơi (Game), từ điển...

Các chương trình có các đặc điểm là:

- Viết bằng Java, phần mở rộng là apk.

- Khi mỗi ứng dụng chạy, nó có một phiên bản Virtual Machine được dựng lên để phục vụ cho nó. Nó có thể là nột Active Program: Chương trình có giao diện với người sử dụng hoặc là một background: chương trình chạy nền hay là dịch vụ.

- Android là hệ điều hành đa nhiệm, điều đó có nghĩa là trong cùng một thời điểm, có thể có nhiều chương trình cùng chạy một lúc, tuy nhiên với mỗi ứng dụng thì có duy nhất một thực thể (instance) được phép chạy mà thôi. Điều đó có tác dụng hạn chế sự lạm dụng tài nguyên, giúp hệ thống hoạt động tốt hơn.

- Các ứng dụng được gán số ID của người sử dụng nhằm phân định quyền hạn khi sử dụng tài nguyên, cấu hình phần cứng và hệ thống.

- Android là một hệ điều hành có tính mở, khác với nhiều hệ điều hành di động khác, Android cho phép một ứng dụng của bên thứ ba được phép chạy nền. Các ứng dụng đó chỉ có một hạn chế nhỏ đó là nó không được phép sử dụng quá 5~10% công suất CPU, điều đó nhằm dể tránh độc quyền trong việc sử dụng CPU.

- Ứng dụng không có điểm vào cố định, không có phương thức main để bắt đầu. [1]

1.5. Quản lý bộ nhớ

Android được thiết kế để quản lý bộ nhớ (RAM) để giảm tối đa tiêu thụ điện năng, trái với hệ điều hành máy tính để bàn luôn cho rằng máy tính sẽ có nguồn điện không giới hạn. Khi một ứng dụng Android không còn được sử dụng, hệ thống sẽ tự động ngưng nó trong bộ nhớ - trong khi ứng dụng về mặt kỹ thuật vẫn "mở", những ứng dụng này sẽ không tiêu thụ bất cứ tài nguyên nào (như năng lượng pin hay năng lượng xử lý) và

nằm đó cho đến khi nó được cần đến. Cách làm như vậy có lợi kép là vừa làm tăng khả năng phản hồi nói chung của thiết bị Android, vì ứng dụng không nhất thiết phải đóng rồi mở lại từ đầu, vừa đảm bảo các ứng dụng nền không làm tiêu hao năng lượng một cách không cần thiết.

Android quản lý các ứng dụng trong bộ nhớ một cách tự động: khi bộ nhớ thấp, hệ thống sẽ bắt đầu diệt ứng dụng và tiến trình không hoạt động được một thời gian, sắp theo thời điểm cuối mà chúng được sử dụng (tức là cũ nhất sẽ bị tắt trước). Tiến trình này được thiết kế ẩn đi với người dùng, để người dùng không cần phải quản lý bộ nhớ hoặc tự tay tắt các ứng dụng. Tuy nhiên, sự che giấu này của hệ thống quản lý bộ nhớ Android đã dẫn đến sự thịnh hành của các ứng dụng tắt chương trình của bên thứ ba trên của hàng Google Play; những ứng dụng kiểu như vậy được cho là có hại nhiều hơn có lợi. [1]

1.6. Cập nhật

Google đưa ra các bản cập nhật lớn cho Android theo chu kỳ từ 6 đến 9 tháng, mà phần lớn thiết bị đều có thể nhận được qua sóng không dây. Bản cập nhật lớn mới nhất là Android 9.0 Pie.

So với các hệ điều hành cạnh tranh khác, Như iOS, các bản cập nhật Android thường mất thời gian lâu hơn để đến với các thiết bị, Với những thiết bị không thuộc dòng Nexus và Pixel, các bản cập nhật thường đến sau vài tháng kể từ khi phiên bản được chính thức phát hành. Nguyên nhân của việc này một phần do sự phong phú về phần cứng của các thiết bị Android, nên người ta phải mất thời gian điều chỉnh bản cập nhật cho phù hợp, vì mã nguồn chính thức của Google chỉ chạy được trên những thiết bị Nexus chủ lực của họ. Chuyển Android sang những phần cứng cụ thể là một quy trình tốn thời gian và công sức của các nhà sản xuất thiết bị, những người luôn ưu tiên các thiết bị mới nhất và thường bỏ rơi các thiết bị cũ hơn. Do đó, những chiếc điện thoại thông minh thế hệ cũ thường không được cập nhật nếu nhà sản xuất quyết định rằng nó không đáng để bỏ thời gian, bất kể chiếc điện thoại đó có khả năng chạy bản cập nhật hay không. Vấn đề này còn trầm trọng hơn khi những nhà sản xuất điều chỉnh Android để đưa giao diện và ứng dụng của họ vào, những thứ này cũng sẽ phải làm lại cho mỗi bản cập nhật. Sự chậm trễ còn được đóng góp bởi nhà mạng, sau khi nhận được bản cập nhật từ nhà sản xuất, họ còn điều chỉnh thêm cho phù hợp với nhu cầu rồi thử nghiệm kỹ lưỡng trên hệ thống mạng của họ trước khi chuyển nó đến người dùng. [1]

1.7. Bảo mật và riêng tư

Các ứng dụng Android chạy trong một "sandbox", là một vực riêng rẽ với hệ thống và không được tiếp cận đến phần còn lại của tài nguyên hệ thống, trừ khi nó được người dùng trao quyền truy cập một cách công khai khi cài đặt. Trước khi cài đặt ứng dung, cửa hàng Play sẽ hiển thị tất cả các quyền mà ứng dụng đòi hỏi: ví dụ như một trò chơi cần phải kích hoạt bộ rung hoặc dữ liệu vào thẻ nhớ SD, nhưng nó không nên cần quyền đọc tin nhắn SMS hoặc tiếp cận danh bạ điện thoại. Sau khi xem xét các quyền này, người dùng có thể chọn đồng ý hoặc từ chối chúng, ứng dụng chỉ được cài đặt khi người dùng đồng ý.

Điện thoại thông minh Android có khả năng báo cáo vị trí của điểm truy cập Wi-Fi, phát hiện ra việc di chuyển của người dùng điện thoại, để xây dựng những cơ sở dữ liệu có chứa vị trí của hàng trăm triệu điểm truy cập. Những cơ sở dữ liệu này tạo nên một bản đồ điện tử để tìm vị trí điện thoại thông minh, cho phép chúng chạy các ứng dụng như Google Latitude, Facebook Places, và gửi những đoạn quảng cáo dựa trên vị trí. [1]

1.8. Ưu và nhược điểm

1.8.1. Ưu điểm

Là hệ điều hành có mã nguồn mở nên khả năng tùy biến cao, có thể tùy ý chỉnh sửa mà không có sự can thiệp hay cấm cản từ Google.

Đa dạng sản phẩm, rất nhiều hãng điện thoại, thiết bị công nghệ đã ưu ái chọn Android cho thiết bị của họ, giá cả thì hợp lý từ bình dân đến cao cấp.

Kho ứng dụng Google Play Store đồ sộ.

Thân thiện và dễ sử dụng.

Khản năng đa nhiệm, chạy cùng lúc nhiều ưng dụng cao.

1.8.2. Nhược điểm

Dễ nhiễm phần mềm độc hại và virus. Do đó tính chất mã nguồn mở, nhiều phần mềm không được kiểm soát có chất lượng không tốt hoặc lỗi bảo mật vẫn được sử dụng.

Sự phân mảnh lớn. Trong khi một số thiết bị Android xuất sắc đã trình làng thì vẫn còn nhiều sản phẩm giá rẻ bình thường khác.

Cập nhật không tự động với tất cả thiết bị. Khi một phiên bản hệ điều hành mới ra mắt, không phải tất cả sản phẩm đều được cập nhật, thậm chí nếu muốn trải nghiệm thì phải mua thiết bị mới.

1.9. Thiết bị

Ngoài điện thoại và máy tính bảng, do bản chất mở và cho phép thay đổi của Android giúp nó xuất hiện trên các thiết bị điện tử khác, như laptop, netbook, smartbook, TV thông minh và máy ảnh (Galaxy Camera). Hơn thế nữa, hệ điều hành Android còn được ứng dụng trong kính mắt thông minh (Project Glass), đồng hồ đeo tay, tai nghe, đầu CD và DVD cho xe hơi, máy nghe nhạc bỏ túi và máy trò chơi điện tử chạy Android. [1]

1.10. Kết luận

Mới bước chân vào làng điện thoại di động nhưng Android đã lần lượt hạ gục những đối thủ tầm cỡ và trở thành hệ điều hành tăng trưởng nhanh nhất trên thế giới. Mỗi phiên bản Android luôn thu hút mối quan tâm lớn của giới công nghệ, các nhà sản xuất và người sử dụng. Thị phần điện thoại Android trên thị trường ngày càng tăng mạnh mẽ, phần nào đó chứng tỏ những ưu thế và tính năng vượt trội của nền tảng này, đồng thời khẳng định những nỗ lực của Android trong việc mạng đến cho người tiêu dùng những sản phẩm tốt nhất.

CHƯƠNG 2
MÔI TRƯỜNG LẬP TRÌNH ANDROID STUDIO

Có nhiều công cụ để phát triển Android nhưng đến nay công cụ chính thức và mạnh mẽ nhất là Android Studio. Đây là môi trường phát triển tích hợp chính thức cho nền tảng Android, được phát triển bởi Google và được sử dụng để tạo phần lớn các ứng dụng mà bạn có thể sử dụng hàng ngày.

Chức năng của Android Studio là cung cấp giao diện để tạo các ứng dụng và xử lý phần lớn các công cụ quản lý file phức tạp đằng sau hậu trường. Ngôn ngữ lập trình được sử dụng ở đây là Java và được cài đặt riêng trên thiết bị của bạn.

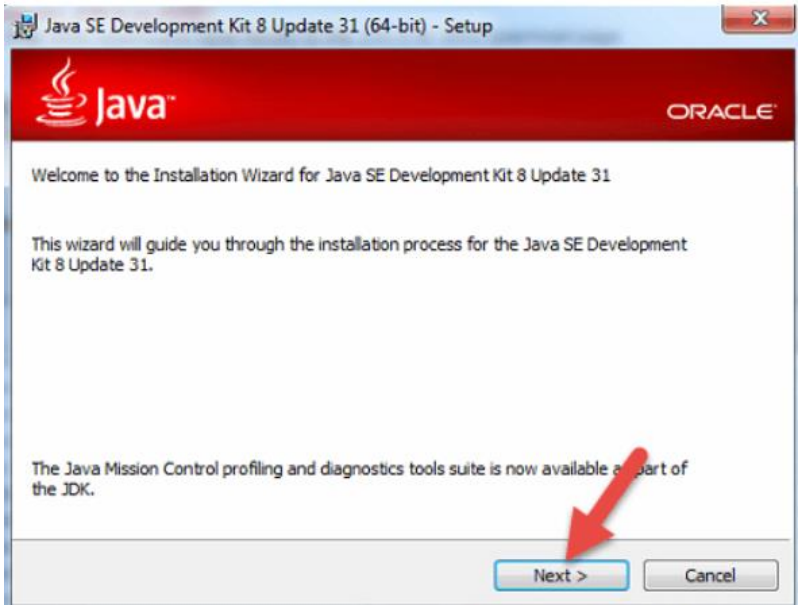
2.1. Thiết lập môi trường

Hai thành phần cơ bản nhất mà chúng ta cần phải có để lập trình Android là bộ JDK (Java Development Kit) và Android SDK (Software Development Kit). JDK dùng để tạo ra môi trường thực thi máy ảo cho hệ điều hành mà chúng ta sử dụng. Android SDK chứa các phiên bản Android, các hàm API cần thiết, source code minh họa cũng như các công cụ hỗ trợ lập trình khác. Mỗi khi Google ra phiên bản Android mới thì Android SDK cũng được cập nhật tương ứng.

2.1.1. Cài đặt Java

Để cài đặt JDK, cần phải truy cập vào trang của Oracle JDK.

https://www.oracle.com/technetwork/java/javase/downloads/index.html



Hình 2.1.1.1: Cài đặt JDK

2.1.2. Cài đặt Android Studio

Truy cập vào trang web chính thức của Android Developer để tải bộ cài đặt của Android Studio.

https://developer.android.com/studio/index.html

Sau khi download xong, nhấn vào file .exe vừa download và kích hoạt để cài đặt. Lúc này chỉ cần thực hiện lần lượt các bước theo hướng dẫn như chọn các thành phần cài đặt (nên chọn hết nếu là cài đặt lần đầu), xác định đường dẫn cần thiết và các thông số khác.

Hình 2.1.2.1: Cài đặt Android Studio

2.2. Các thành phần trong một dự án Android

2.2.1. File manifest

Trong bất kì một dự án Android nào khi tạo ra đều có một file manifest, file này được dùng để đăng ký các màn hình sử dụng trong ứng dụng Android, 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.

Dưới đây là nội dung của một file AndroidManifest.xml

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

  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
```

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

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

Application

Thẻ <application>, bên trong thẻ này chứa các thuộc tính được định nghĩa cho ứng dụng Android như:

- **android:icon** = "drawable resource". Ở đây đặt đường dẫn đến file icon của ứng dụng khi cài đặt. VD: android:icon = "@mipmap/ic_launcher".

- **android:name** = "string". Thuộc tính này để đặt tên cho ứng dụng Android. Tên này sẽ được hiển thị lên màn hình sau khi cài đặt ưng dụng.

- **android:theme** = "drawable theme". Thuộc tính này để đặt theme cho ứng dụng. Các theme là các cách để hiển thị giao diện ứng dung.

Ngoài ra còn nhiều thuộc tính khác....

2.2.2. File java

File java là một file tự động sinh ra ngay khi tạo ứng dụng, file này đực sử dụng để quản lý thuộc tính được khai báo trong file XML của ứng dụng và các tài nguyên hình ảnh. Mã nguồn của file java được tự đọng sinh ra khi có bất kì một sự kiện này xảy ra làm thay đổi các thuộc tính trong ứng dụng.

Chẳng hạn như, kéo và thả một file hình ảnh từ bên ngoài vào project thì ngay lập tức thuộc tính đường dẫn đến file đó cũng sẽ được hình thành trong file java hoặc xóa một file hình ảnh thì đường dẫn tương ứng đến hình ảnh đó cũng tự động bị xóa.

- Dưới đây là ví dụ về nội dung của một file java:

```
package com.example.tonghaisointerface;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
   EditText edt_so1, edt_so2, edt_kq;
   Button btn_tinh;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      edt_so1 = findViewById(R.id.edt_so1);
      edt_so2 = findViewById(R.id.edt_so2);
      edt_kq = findViewById(R.id.edt_kq);
      btn_tinh = findViewById(R.id.btn_tinh);
      btn_tinh.setOnClickListener(this);
   }
   @Override
   public void onClick(View view) {
            String chuoi1 = edt_so1.getText().toString();
            String chuoi2 = edt_so2.getText().toString();
            int so1 = Integer.parseInt(chuoi1);
            int so2 = Integer.parseInt(chuoi2);
            int kq = so1 + so2;
            edt_kq.setText(String.valueOf(kq));
   }
}
```

2.2.3. File tài nguyên

Chứa các tài nguyên mà ứng dụng sẽ sử dụng đến, nó tổ chức thành các thư mục con như:

- drawable/ : ở đây cơ bản lưu các đối tượng đồ họa như các ảnh dạng png, các ảnh dạng xml...

- layout/: lưu trữ các file xml biểu diễn về thành phần, bố cục của các thành phần hiển thị được trên màn hình.

- mipmap/: cũng để lưu các đối tượng hình ảnh, ví dụ icon ứng dụng ic_launcher đặt ở đây.

- values/: chứa các file như color.xml, dimens.xml, string.xml, style.xml, đây là các file xml định nghĩa các giá trị có thể sử dụng trong ứng dụng như màu sắc, kích thước, các chuỗi, các theme ...

2.2.4. File Grandle

Gradle Scrips: Chứa nhiều nhánh con như build.gradle, local.properties, ... là nơi bạn thiết lập các thông số để Gradle build ứng dụng. Lưu ý Gradle là một công cụ tích hợp vào Android Studio, chức năng của nó build mã nguồn, kết hợp tài nguyên, phân tích xml ... rồi kết hợp chúng lại với nhau tạo ra ứng dụng chạy trên JVM.

2.3. Chu kỳ ứng dụng Android

Một tiến trình Linux gói gọn một ứng dụng Android đã được tạo ra cho ứng dụng khi code cần được run và sẽ còn chạy cho đến khi:

- Nó không phụ thuộc.

- Hệ thống cần lấy lại bố nhớ mà nó chiếm giữ cho các ứng dụng khác.

Một sự khác thường và đặc tính cơ bản của Android là thời gian sống của tiến trình ứng dụng không được điều khiển trực tiếp bởi chính nó. Thay vào đó, nó được xác định bởi hệ thống qua một kết hợp của:

- Những phần của ứng dụng mà hệ thống biết đang chạy.

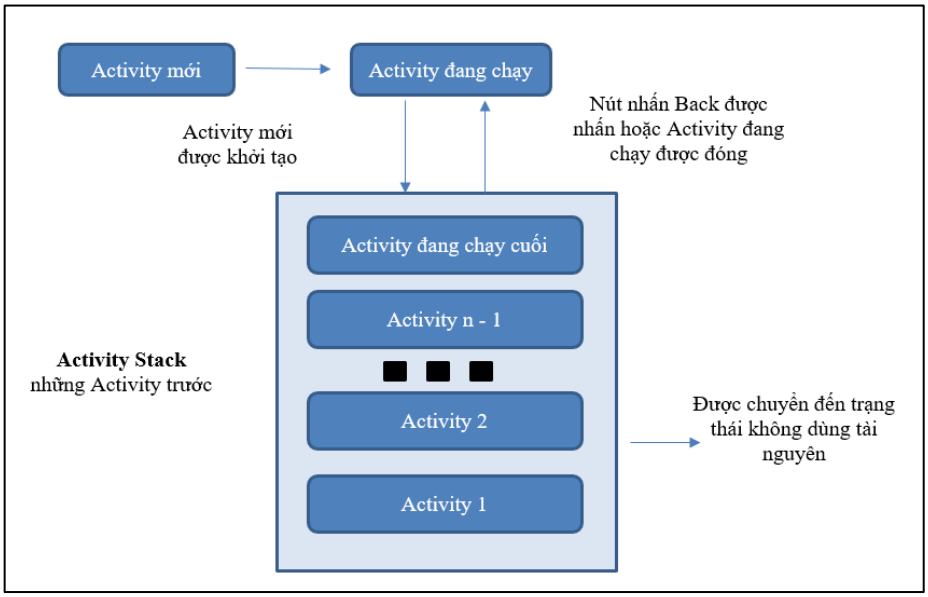
- Những phần quan trọng như thế nào đối với người dùng.

2.3.1. Chu kỳ sống thành phần

Các thành phần ứng dụng có một chu kỳ sống, tức là mỗi thành phần từ lúc bắt đầu khởi tạo và đến thời điểm kết thúc. Giữa đó, đôi lúc chúng có thể là active hoặc inactive, hoặc là trong trường hợp actives nó có thể visible hoặc invisible.

2.3.2. Ngăn xếp Activity

Bên trong hệ thống các activity được quản lý như một ngăn xếp Activity. Khi một Activity mới được bắt đầu, nó được đặt ở đỉnh của ngăn xếp và trở thành activity đang chạy.

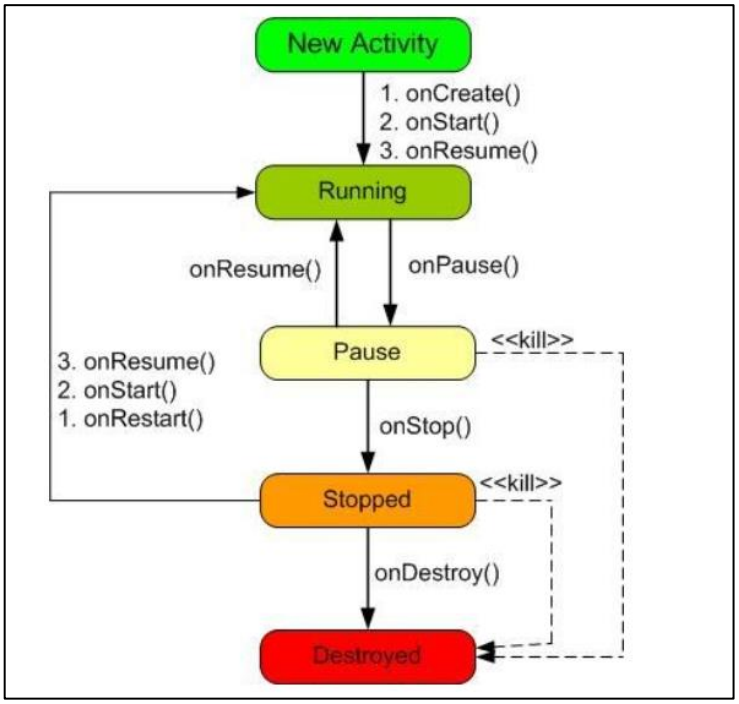


Hình 2.3.2.1: Ngăn xếp Activity

Activity trước sẽ ở bên dưới activity mới và sẽ không thấy trong suốt quá trình activity mới tồn tại.

Nếu người dùng nhấn nút Back thì activity kế tiếp của ngăn xếp sẽ di chuyển lên và trở thành active.

2.3.3. Vòng đời của Activity



Hình 2.3.3.1: Vòng đời của Activity

Một Activity chủ yếu có 3 chu kỳ chính sau:

- Running: khi Activity là được chạy trên màn hình. Activity này tập trung vào những thao tác của người dùng trên ứng dụng.

- Paused: Activity là được tạm dừng khi mất focus nhưng người dùng vẫn trông thấy. Có nghĩa là một Activity mới ở trên nó nhưng không bao phủ đầy màn hình. Một Activity tạm dừng là còn sống nhưng có thể bị kết thúc bởi hệ thống trong trường hợp thiếu vùng nhớ.

- Stopped: Nếu nó hoàn toàn bao phủ bởi Activity khác. Nó vẫn còn trạng thái và thông tin trong nó. Người dùng không thấy nó và thường bị loại bỏ trong trường hợp hệ thống cần vùng nhớ cho tác vụ khác. [2]

2.3.4. Các phương thức của chu kỳ sống

Phương thức: onCreate() :

- Được gọi khi activity lần đầu tiên được tạo.

- Ở đây bạn làm tất cả các cài đặt tĩnh - tạo các view, kết nối dữ liệu đến list ....

- Phương thức này gửi qua nột đối tượng *Bundle* chứa đựng từ trạng thái trước của Activity.

- Luôn theo sau bởi *onStart().*

Phương thức: onStart() :

- Được gọi trước khi một Activity visible với người dùng.

- Theo sau bởi *onResume()* nếu activity đến trạng thái foreground hoặc *onStop()* nếu nó trở nên ẩn.

Phương thức: onResume() :

- Được gọi trước khi activity bắt đầu tương tác với người dùng.

- Tại thời điểm này activity ở trên đỉnh của stack activity.

- Luôn theo sau bởi *onPause().*

Phương thức: onPause() :

- Được gọi khi hệ thống đang resuming activity khác.

- Phương thức này là điển hình việc giữ lại không đổi dữ liệu.

- Nó nên được diễn ra một cách nhanh chóng bởi vì activity kế tiếp sẽ không được resumed ngay cho đến khi nó trở lại.

- Theo sau bởi *onResume()* nếu activity trở về từ ở trước, hoặc bởi *onStop()* nếu nó trở nên visible với người dùng.

- Trạng thái của activity có thể bị giết bởi hệ thống.

Phương thức: onStop() :

- Được gọi khi acitivity không thuộc tầm nhìn của người dùng.

- Nó có thể diễn ra bởi vì nó đang bị hủy, hoặc bởi vì activity khác vừa được resumed và bao phủ nó.

- Được theo sau bởi *onRestart()* nếu activity đang mở lại để tương tác với người dùng, hoặc *onDestroy()* nếu activity đang bỏ.

- Trạng thái của activity có thể bị giết bởi hệ thống.

Phương thức: onDestroy() :

- Được gọi trước khi activity bị hủy.

- Đó là lần gọi cuối cùng mà activity này được nhận.

- Nó được gọi khác bởi vì activity đang hoàn thành, hoặc bởi vì hệ thống tạm thời bị hủy diệt để tiết kiệm vùng nhớ.

- Bạn có thể phân biệt giữa 2 kịch bản với phương thức *isFinishing()*.

- Trạng thái của activity có thể bị giết bởi hệ thống.

2.4. View

Trong một ứng dụng Android, giao diện người dùng được xây dựng từ các đối tượng View và View Group. Có nhiều kiểu View và View Group, mỗi một kiểu là một hậu duệ của class View và 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.

2.4.1. TextView

TextView là một View cho phép hiển thị các dòng chữ (text) trên màn hình, nó có nhiều thuộc tính tùy mục đích sử dụng mà áp dụng, như thiết lập cỡ chữ, font chữ, màu chữ.

Khai báo TextView trong XML:

```
<TextView
   android:id="@+id/textview"
   android:text="Hello World!"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

Lấy TextView trong Java:

```
TextView textView;
textView = findViewById(R.id.textview);
```

2.4.2. Button

Button là một loại View, nó hiển thị nút bấm để chờ người dùng bấm vào. Button kế thừa từ TextView nên các thuộc tính, thiết lập cho TextView là có hiểu quả như đối với Button.

Khai báo Button trong XML:

```
<Button
   android:id="@+id/btn_click"
   android:text="Click"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

Lấy Button và bắt sự kiện Click:

```
Button button;
button = findViewById(R.id.btn_click);
button.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View view) {
   // viết code ở đây }
});
```

2.4.3. ImageButton

Cũng tương tự như Button, ImageButton chỉ có thêm thuộc tính

anroid:src = "@drawable/ic_launcher_background" để thêm hình ảnh vào và không có thẻ text.

Khai báo ImageButton trong XML

```
<ImageButton
   android:id="@+id/imb_click"
   android:src="@drawable/ic_launcher_background"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

2.4.4. ImageView

Được dùng để hiển thị tài nguyên hình ảnh như các ảnh Bipmap, cá ảnh Drawable. Nó cung cấp các chức năng tùy biến khác nhau như co kéo/cắt ảnh khi hiển thị trên View.

Khai báo trong ImageView trong XML

```
<ImageView
   android:id="@+id/imageview"
   android:src="@drawable/ic_launcher_background"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

2.4.5. EditText

EditText là loại View hiển thị một hộp (chữ nhật) cho phép người dùng nhập dữ liệu (chữ, số ... có thể khống chế nhập dữ liệu là text, số, phone, ngày tháng ...).

Do EditText mở rộng chức năng từ TextView, nên các tùy chọn thiết lập trình bày ở TextView vẫn đúng cho EditText.

```
<EditText
   android:id="@+id/edittext"
   android:text="Hello World!"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```

2.4.6. ListView

Được sử dụng để thể hiện một danh sách các thông tin theo từng hàng. Một hàng thông thường được load lên từ một file XML đã được cố định trên đó số lượng thông tin và

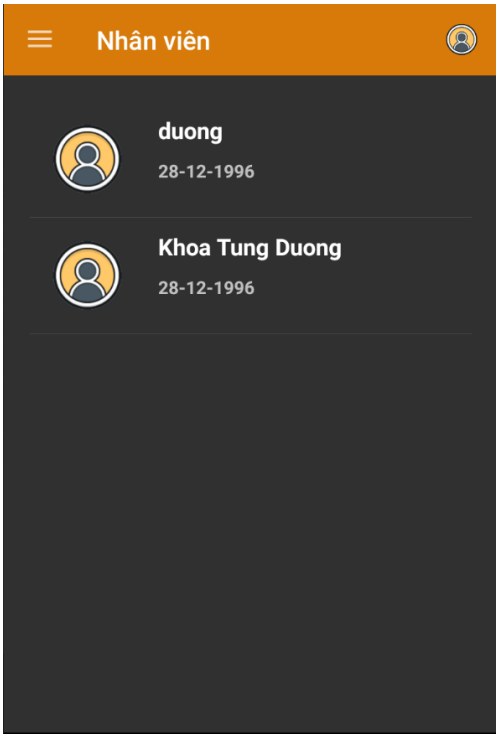
loại thông tin cần được thể hiện.

Để thể hiện được một list thông tin trên một screen thì cần phải có 3 yếu tố chính:

- Data Source: Data Source có thể là một ArrayList hoặc bất kỳ một cấu trúc dữ liệu kiểu danh sách nào.

- Adapter: Adapter là một class trung gian giúp ánh xạ dữ liệu trong Data Source vào đúng vị trí hiển thị trong ListView. Chẳng hạn, trong Data Source có một trường name và trong ListView cũng có một TextView để thể hiện trường name này. Tuy nhiên ListView sẽ không thể hiển thị dữ liệu trong Data Source lên được nếu như Adapter không gán dữ liệu vào cho đối tượng hiển thị.

- ListView: ListView là đối tượng để hiển thị các thông tin trong Data Source ra một cách trực quan và người dùng có thể thao tác trực tiếp trên đó.



Hình 2.4.6.1: Hình minh họa ListView

Khai báo ListView trong XML:

```
<ListView
   android:id="@+id/listview"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
</ListView>
```

Gán cho ListView một Adapter là nguồn cấp dữ liệu cho nó.

```
listView = findViewById(R.id.listview);
listView.setAdapter(ListViewAdapter);
```

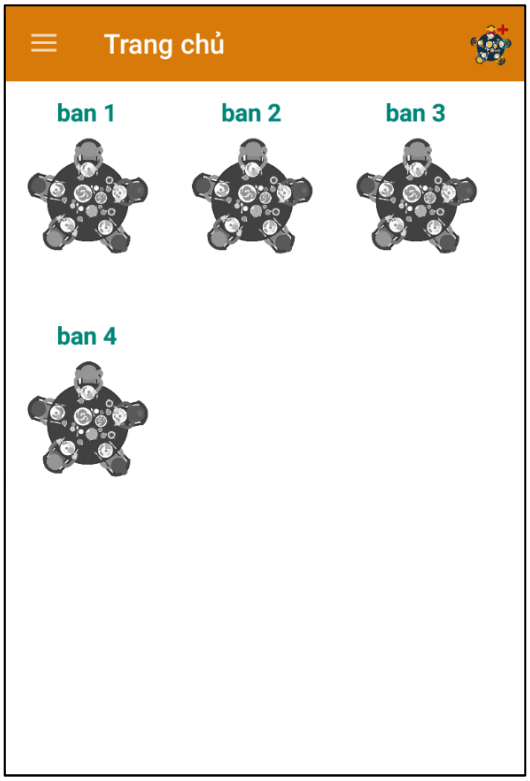
Khi đã gán Adapter vào ListView, thì ListView sẽ dùng Adapter này xác định cần hiển thị bao nhiêu phần tử, mỗi phần tử có View như thế nào do Adapter tạo và đính vào ListView, mỗi khi dữ liệu Adapter quản lý thay đổi , cần thông báo cho ListView biết mà cập nhật bằng cách gọi:

```
ListViewAdapter.notifyDataSetChanged();
```

2.4.7. GridView

GridView cũng tương tự như ListView, GridView cũng dựa vào DataSource, và Adapter.

Điểm khác nhau là GridView có thể thiết lập số cột. Dữ liệu luôn đưa vào dưới dạng mảng, list một chiều, nhưng dựa vào số cột ta thiết lập nó tự động ngắt hàng.



Hình 2.4.7.1: Hình minh họa GridView

Khai báo GridView trong XML.

```
<GridView
   android:numColumns="3"
   android:id="@+id/gridview"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
</GridView>
```

Thiết lập *android:numColumns="3"*, tức là GridView sẽ ngắt dòng khi đủ 3 phần tử, nó chỉ khác chỗ này, còn việc đưa dữ liệu lên giống hệt như làm với ListView. [2]

2.5. ViewGroup

Một ViewGroup là một đối tượng sử dụng để chứa các đối tượng View và GroupView 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 do đó có thể tạo các layout phức tạp hơn.

2.5.1. LinearLayout

LinearLayout được dùng để bố trí các thành phần giao diện theo chiều ngang hoặc chiều dọc nhưng trên một line duy nhất mà không có xuống dòng.

LinearLayout làm cho các thành phần trong nó không bị phụ thuộc vào kích thước của màn hình. Các thành phần trong LinearLayout được dàn theo những tỉ lệ cân xứng dựa vào các ràng buộc giữa các thành phần.

2.5.2. RelativeLayout

RelativeLayout là một layout mà nó thực hiện các view con nó chứa ở các vị trí trong mối liên hệ của chúng với nhau (như View con này nằm dưới một View con khác View con này nằm bên trái một View con khác), kể cả mối liên hệ của chúng với chính phần tử cha RelativeLayout (như căn thẳng theo cạnh đáy của phần tử cha, nằm giữa phần tử cha, nằm bên trái phần tử cha).

2.5.3. FrameLayout

FrameLayout là loại View cơ sở, nó là loại Layout đơn giản nhất. Mặc dù nó có thể chứa nhiều View con bên trong, nhưng mục đích chính thiết kế ra nó để chứa một View, từ đó nó trở thành cơ sở để tạo ra các View khác phức tạp hơn. Khi thiết kế Layout chứa nhiều View thì không nên sử dụng layout này, vì nó quá đơn giản việc bố cục các View con trong nó rất khó khăn (nó không có các tính năng điều khiển vị trí View con sao cho việc độc lập về màn hình được đảm bảo).

2.5.4. TableLayout

Layout này thường được sử dụng khi cần thiết kế một table chứa dữ liệu hoặc cần bố trí các widget theo các hàng và cột. Chẳng hạn như, giao diện của một chiếc máy tính đơn giản. [2]

2.6. Activity & Intent

2.6.1. Activity

Activity là một thành phần chính của một ứng dụng Android, được dùng để hiển thị một màn hình và nắm bắt các hoạt động xảy ra trên màn hình đó. Khi làm việc với Activity cần nắm bắt được một số kiến thức cơ bản sau:

- Vòng đời của một ứng dụng (*xem vòng đời của một ứng dụng Android mục 2.3.3*).

- Tạo menu và dialog.

- Khởi động một Activity: Để khởi động một Activity ta sử dụng Intent (*Mục 2.6.2*).

Ví dụ: Từ Activity A muốn chuyển qua Activity B ta khai báo một Intent trong Activity A:

```
Intent intent = new Intent(A.this, B.class);
startActivity(intent);
```

- Tính liên lạc giữa 2 activity.

Khi chuyển sang một Activity khác ta có thể gửi kèm dữ liệu trong intent đó như sau:

```
intent.putExtra("vidu", vidu);
```

Bên phía Activity được khởi động hay được chuyển đến, có thể lấy dữ liệu được gửi như sau:

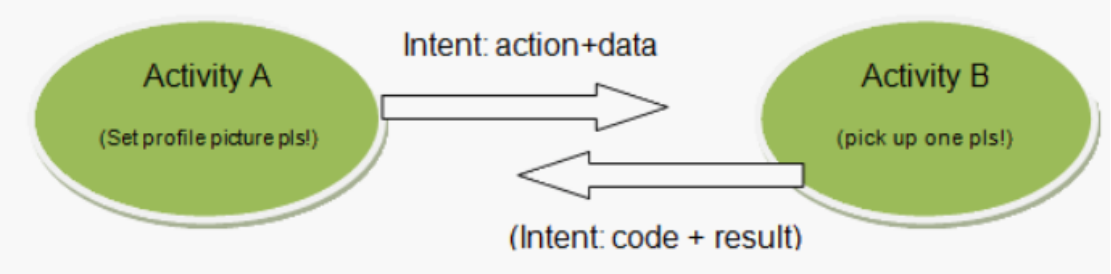
```
getIntent().getIntExtra("vidu",0);
```

2.6.2. Intent

Khái niệm Intent:

- Là một cấu trúc dữ liệu mô tả cách thức, đối tượng thực hiện của một Activity.

- Là cầu nối giữa các Activity : ứng dụng Android thường bao gồm nhiều Activity, mỗi Activity hoạt động độc lập với nhau và thực hiện những công việc khác nhau. Intent chính là người đưa thư, giúp các Activity có thể triệu gọi cũng như truyền các dữ liệu cần thiết tới một Activity khác.



Hình 2.6.2.1: Truyền dữ liệu giữa 2 Activity

Code khởi chạy Activity thông thường:

```
// gọi đến Activity cần chạy
Intent intent = new Intent(A.this, B.class);
// nếu có truyền tham số cho Activity
intent.putExtra("dulieu", 123);
// Gửi Intent cho hệ thống Android để kích hoạt Activity
startActivity(intent);
```

CHƯƠNG 3
KỸ THUẬT XÂY DỰNG ỨNG DỤNG

3.1. Hệ quản trị cơ sở dữ liệu SQLite

SQLite là một hệ quản trị cơ sở dữ liệu tương tự như MySQL. Đặc điểm của SQLite là gọn, nhẹ, đơn giản. Chương trình gồm 1 file chưa đến 500KB, không cần cài đặt, không cần cấu hình hay khởi động mà có thể sử dụng ngay. Dữ liệu database cũng được lưu ở một file duy nhất. Không có khái niệm user, password hay quyền hạn trong SQLite database.

SQLite không thích hợp với những hệ thống lớn nhưng ở quy mô vừa tầm thì SQLite phát huy uy lực và không hề yếu kém về mặt chức năng hay tốc độ. Với các đặc điểm trên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm và là sự lựa chọn phù hợp cho những người bắt đầu học database.

Hiện nay thì SQLite đã được ứng dụng vào smartphone như iOS và Android để lưu trữ dữ liệu.

3.2. Các thao tác cơ bản trên cơ sở dữ liệu

3.2.1. Tạo cơ sở dữ liệu

Để tạo một cơ sở dữ liệu trong SQLite, đầu tiên ta tạo một lớp kế thừa lớp SQLiteOpenHelper dùng để thao tác với cơ sở dữ liệu. Trong đó, ClassName là tên lớp.

```
public class ClassName extends SQLiteOpenHelper {
// Tạo phương thức khởi tạo
// Override phương thức onCreate
public void onCreate (SQLiteDatabase db) {
        String sql = "câu lệnh tạo bảng";
        db.execSQL(sql);
}
/* Các phương thức:
thêm dữ liệu, cập nhật dữ liệu,
xóa dữ liệu và truy vấn dữ liệu */
}
```

Sau đó, ta cần tạo phương thức khởi tạo là một lớp con của lớp SQLiteOpenHelper, được truyền vào 4 tham số context, name, factory, version.

```
public ClassName(Context context) {
   super(context, name, factory, version);
}
```

- *context*: là giao diện chứa thống tin toàn cục về môi trường ứng dụng.

- *name*: Tên của database.

- *factory*: thường để mặc định là null.

- *version*: phiên bản của database, với phiên bản đầu tiên chúng ta để giá trị bằng 1.

Khi khởi tạo một đối tượng của lớp này, ta phải ghi đè 2 phương thức *onCreate()* và *onUpgrade().*

- *onCreate**():*** được gọi khi cơ sở dữ liệu được tạo lần đầu.

- *onUpgrade():* được gọi khi ứng dụng chứa nhiều phiên bản của cơ sở dữ liệu.

Và có 2 phương thức *getWritableDatabase(), getReadableDatabase()* được sử dụng để đọc và ghi dữ liệu:

- *getWritableDatabase():* mở hay tạo một cơ sở dữ liệu cho việc đọc và ghi. Trả về một tham chiếu đến một cơ sở dữ liệu trong hình thức một đối tượng SQLiteDatabase.

- *getReadableDatabase():* mở hay tạo một cơ sở dữ liệu chỉ cho việc đọc. Trả về một tham chiếu đến một cơ sở dữ liệu trong hình thức một đối tượng SQLiteDatabase.

3.2.2. Các thao tác trên cơ sở dữ liệu

a. Tạo bảng trong SQLite

Để tạo bảng trong cơ sở dữ liệu, chúng ta dùng phương thức execSQL của lớp SQLiteDatabase để thực hiện lệnh truy vấn tại bảng. Lệnh SQL tạo bảng sẽ được chuyển thành chuỗi để chuyển đến phương thức *execSQL().*

```
public void onCreate (SQLiteDatabase db) {
    String sql = "CREATE TABLE" table_name "(
                " trường_1 " kiểu_dữ_liệu,
                " trường_2 " kiểu_dữ_liệu,
                ...
                " trường_n " kiểu_dữ_liệu )";
    db.execSQL(sql);
   }
```

b. Thêm dữ liệu

Sử dụng phương thức *insert* để tạo mới một bản ghi vào cơ sở dữ liệu.

Ví dụ: ta có một đối tượng db thuộc kiểu SQLiteDatabase.

Đầu tiên gọi phương thức getWritableDatabase mở cơ sở dữ liệu lên để đọc và ghi. Sau đó tạo một đối tượng thuộc ContentValues để có thể put các giá trị vào trong các trường. Sau khi đã put hết giá trị rồi chúng ta gọi phương thức *insert()* và truyền vào 3 tham số:

- *Tham số 1*: tên bảng.

- *Tham số 2*: thường để mặc định là null.

- *Tham số 3*: đối tượng cần thêm.

```
public long create (Kiểu_dữ_liệu giá_trị){
        SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put( tên trường , giá trị);
    db.insert( tên bảng , null , contentValues );
  }
```

c. Cập nhật dữ liệu

Phương thức *update* của SQLiteDatabase để cập nhật dữ liệu trong bảng theo một điều kiện bất kỳ nào đó. Phương thức này trả về số dòng bị ảnh hưởng. Ví dụ nếu có 3 dòng bị thay đổi thì nó trả về 3, nếu không có dòng nào bị ảnh hưởng thì nó trả về 0.

Đầu tiên gọi phương thức getWritableDatabase mở cơ sở dữ liệu lên để đọc và ghi. Sau đó tạo một đối tượng thuộc ContentValues để có thể put các giá trị vào trong các trường.

Sau đã put hết giá trị rồi chúng ta gọi phương thức *update()* và truyền vào 4 tham số:

- *Tham số 1*: tên bảng.

- *Tham số 2*: đối tượng muốn chỉnh sửa với giá trị mới.

- *Tham số 3*: tập các điều kiện lọc.

- *Tham số 4*: tập các giá trị của điều kiện lọc, thường để mặc định là null.

```
public long update (Kiểu_dữ_liệu giá_trị) {
        SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put( tên trường , giá trị);
        String where = tên_trường " = " giá_trị ;
    db.update( tên bảng , contentValues , where , whereArgs );
 }
```

d. Xóa dữ liệu

Chúng ta sử dụng phương thức *delete* của SQLiteDatabase để xóa dữ liệu trong bảng theo một điều kiện bất kỳ nào đó. Phương thức này trả về số dòng bị ảnh hưởng. Muốn xóa toàn bộ dữ liệu trong bảng thì ta truyền null vào 2 đối số cuối.

- *Tham số 1*: tên bảng.

- *Tham số 2*: tập điều kiện lọc.

- *Tham số 3*: tập các giá trị của điều kiện lọc, thường được để mặc định là null.

```
public long delete (Kiểu_dữ_liệu giá_trị){
            SQLiteDatabase db = this.getWritableDatabase();
            String where = tên_trường " = " giá_trị ;
            db.delete( tên bảng , where , whereArgs );
   }
```

f. Lấy danh sách dữ liệu

Để lấy danh sách dữ liệu, ta dùng phương thức *rawQuery* được dùng để thực hiện truy vấn và trả về kết quả phù hợp thông qua một đối tượng Cursor.

Đối tượng cursor được hiểu đơn giản là một con trỏ, trỏ đến kết quả trả về của câu truy vấn, con trỏ này trỏ đến cái bảng trả về của câu truy vấn.

```
public void Name ( tham số ) {
            SQLiteDatabase db = this.getWritableDatabase();
            String sql = " SELECT * [hoặc các trường] FROM " tên_bảng;
            Cursor cursor = db.rawQuery( sql , null );
            cursor.moveToFirst();
            while (! cursor.isAfterLast()) {
             // truy vấn dữ liệu sử dụng
                        cursor.getString(cursor.getcolumnIndex( tên trường ));
                        hoặc cursor.getInt(cursor.getcolumnIndex( tên trường ));
            cursor.moveToNext();
            }
}
```

- *moveToFirst()*: di chuyển đến hàng đầu tiên từ tập kết quả.

- *moveToNext()*: di chuyển đến hàng kế tiếp từ tập kết quả.

- *isAfterLast()*: kiểm tra đã đến cuối kết quả chưa.

- *get<type>()*: trả về giá trị kiểu <type> tại cột xác định của hàng tại vị trí con trỏ hiện tại. Các phương thức phổ biến gồm *getString()*, *getInt()*.

3.3. Quản lý người dùng

Việc quản lý nhân viên trên ứng dụng là điều cần thiết cho mỗi nhà hàng, nó giúp cho chủ nhà hàng có thể quản lý tất cả nhân viên đang làm việc tại nhà hàng.

3.3.1. Thực hiện đăng ký người dùng

Khi nhà hàng có thêm một nhân viên mới, chủ nhà hàng thực hiện chức năng thêm nhân viên.

```
public long ThemNhanVien(NhanVienDTO nhanVienDTO){
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_NHANVIEN_TENDN,nhanVienDTO.getTENDN());
    contentValues.put(CreateDatabase.TB_NHANVIEN_MATKHAU,nhanVienDTO.getMATKHAU());
    contentValues.put(CreateDatabase.TB_NHANVIEN_GIOITINH,nhanVienDTO.getGIOITINH());
    contentValues.put(CreateDatabase.TB_NHANVIEN_NGAYSINH,nhanVienDTO.getNGAYSINH());
    contentValues.put(CreateDatabase.TB_NHANVIEN_CMND,nhanVienDTO.getCMND());

    long kiemtra = database.insert(CreateDatabase.TB_NHANVIEN, nullColumnHack: null,contentValues);
    return kiemtra;
}
```

3.3.2. Thay đổi thông tin người dùng

Khi thông tin nhân viên đang làm trong nhà hàng thông tin chưa đủ, việc cập nhật thông tin nhân viên là điều cần thiết.

```
public boolean SuaNhanVien(NhanVienDTO nhanVienDTO){
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_NHANVIEN_TENDN,nhanVienDTO.getTENDN());
    contentValues.put(CreateDatabase.TB_NHANVIEN_MATKHAU,nhanVienDTO.getMATKHAU());
    contentValues.put(CreateDatabase.TB_NHANVIEN_GIOITINH,nhanVienDTO.getGIOITINH());
    contentValues.put(CreateDatabase.TB_NHANVIEN_NGAYSINH,nhanVienDTO.getNGAYSINH());
    contentValues.put(CreateDatabase.TB_NHANVIEN_CMND, nhanVienDTO.getCMND());
    String where = CreateDatabase.TB_NHANVIEN_MANV + " = " + nhanVienDTO.getMANV();

    long kiemtra = database.update(CreateDatabase.TB_NHANVIEN, contentValues,where , whereArgs: null);

    if(kiemtra !=0 ){
        return true;
    }else{
        return false;
    }
}
```

3.3.3. Xóa thông tin người dùng

Khi nhân viên không còn làm trong nhà hàng nữa, chủ nhà hàng chỉ việc thực hiện chức năng xóa nhân viên.

```
public boolean XoaNhanVienTheoMa(int manv){

    long kiemtra = database.delete(CreateDatabase.TB_NHANVIEN,
            whereClause: CreateDatabase.TB_NHANVIEN_MANV + " = " + manv, whereArgs: null);
    if(kiemtra !=0 ){
        return true;
    }else{
        return false;
    }
}
```

3.3.4. Lấy danh sách người dùng

Việc xem danh sách tất cả nhân viên giúp cho chủ nhà hàng có thể biết được hiện tại những nhân viên nào đang làm việc trong nhà hàng.

```
public List<NhanVienDTO> LayDanhSachNhanVien(){
    List<NhanVienDTO> nhanVienDTOs = new ArrayList<NhanVienDTO>();
    String truyvan = "SELECT * FROM " + CreateDatabase.TB_NHANVIEN;
    Cursor cursor = database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        NhanVienDTO nhanVienDTO = new NhanVienDTO();
        nhanVienDTO.setMANV(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_MANV)));
        nhanVienDTO.setTENDN(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_TENDN)));
        nhanVienDTO.setMATKHAU(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_MATKHAU)));
        nhanVienDTO.setGIOITINH(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_GIOITINH)));
        nhanVienDTO.setNGAYSINH(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_NGAYSINH)));
        nhanVienDTO.setCMND(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_CMND)));

        nhanVienDTOs.add(nhanVienDTO);
        cursor.moveToNext();
    }

    return nhanVienDTOs;
}
```

3.3.5. Kiểm tra đăng nhập

Việc thực hiện đăng nhập tài khoản rất quan trọng, để tránh việc không phải là nhân viên của nhà hàng thì không thể đăng nhập được vào ứng dụng.

```
public int KiemTraDangNhap(String tendangnhap, String matkhau){
    String truyvan = "SELECT * FROM " + CreateDatabase.TB_NHANVIEN + " WHERE "
            + CreateDatabase.TB_NHANVIEN_TENDN + " = '"
            + tendangnhap + "' AND " + CreateDatabase.TB_NHANVIEN_MATKHAU + " = '" + matkhau + "'";

    int manhanvien = 0;
    Cursor cursor =  database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        manhanvien = cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_NHANVIEN_MANV));
        cursor.moveToNext();
    }
    return manhanvien;
}
```

3.4. Quản lý bàn ăn

3.4.1. Thêm bàn ăn

Một nhà hàng luôn luôn phải có nhiều bàn ăn, việc nhiều bàn ăn là cần thiết, chủ nhà hàng thực hiện chức năng thêm bàn ăn.

```
public boolean ThemBanAn(String tenban) {
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_BANAN_TENBAN, tenban);
    contentValues.put(CreateDatabase.TB_BANAN_TINHTRANG, "false");

    long kiemtra = database.insert(CreateDatabase.TB_BANAN,  nullColumnHack: null, contentValues);
    if (kiemtra != 0) {
        return true;
    } else {
        return false;
    }
}
```

3.4.2. Thay đổi tên bàn ăn

Khi nhà hàng cần sắp xếp lại vị trí các bàn trong nhà hàng, theo đó cần phải thay đổi tên bàn theo đúng vị trí cần sắp xếp lại.

```
public boolean CapNhatLaiTenBan(int maban, String tenban) {
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_BANAN_TENBAN,tenban);

    long kiemtra = database.update(CreateDatabase.TB_BANAN, contentValues,
            whereClause: CreateDatabase.TB_BANAN_MABAN + " = '" + maban + "'",  whereArgs: null);

    if (kiemtra != 0) {
        return true;
    } else {
        return false;
    }
}
```

3.4.3. Xóa bàn ăn

Khi nhà hàng muốn bỏ vị trí một bàn ăn nào đó, chức năng xóa bàn ăn giúp cho chủ nhà hàng thuận tiện xử lý.

```
public boolean XoaBanAnTheoMa(int maban){
    long kiemtra = database.delete(CreateDatabase.TB_BANAN,
            whereClause: CreateDatabase.TB_BANAN_MABAN + " = " + maban, whereArgs: null);
    if (kiemtra != 0) {
        return true;
    } else {
        return false;
    }
}
```

3.4.4. Lấy danh sách bàn ăn

Xem toàn bộ danh sách bàn ăn giúp cho chủ nhà hàng quản lý được hoạt động của từng bàn ăn.

```
public List<BanAnDTO> LayTatCaBanAn() {
    List<BanAnDTO> banAnDTOList = new ArrayList<~>();
    String truyvan = "SELECT * FROM " + CreateDatabase.TB_BANAN;
    Cursor cursor = database.rawQuery(truyvan,  selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()) {
        BanAnDTO banAnDTO = new BanAnDTO();
        banAnDTO.setMaBan(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_BANAN_MABAN)));
        banAnDTO.setTenBan(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_BANAN_TENBAN)));

        banAnDTOList.add(banAnDTO);
        cursor.moveToNext();
    }
    return banAnDTOList;
}
```

3.5. Quản lý thực đơn

3.5.1. Thêm loại món ăn

Thêm loại món ăn là điều cần thiết, không có loại món ăn thì sẽ không có món ăn nào. Chủ nhà hàng thực hiện chức năng thêm loại món ăn.

```
public boolean ThemLoaiMonAn(String tenloai){
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_LOAIMONAN_TENLOAI,tenloai);

    long kiemtra = database.insert(CreateDatabase.TB_LOAIMONAN, nullColumnHack: null,contentValues);
    if (kiemtra != 0){
        return true;
    }else {
        return false;
    }
}
```

3.5.2. Lấy danh sách loại món ăn

Xem toàn bộ danh sách loại món ăn giúp cho chủ nhà hàng quản lý được các loại món ăn hiện có trong nhà hàng.

```
public List<LoaiMonAnDTO> LayDanhSachLoaiMonAn(){
    List<LoaiMonAnDTO> loaiMonAnDTOS = new ArrayList<~>();

    String truyvan = "SELECT * FROM " + CreateDatabase.TB_LOAIMONAN;
    Cursor cursor = database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        LoaiMonAnDTO loaiMonAnDTO = new LoaiMonAnDTO();
        loaiMonAnDTO.setMaLoai(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_LOAIMONAN_MALOAI)));
        loaiMonAnDTO.setTenLoai(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_LOAIMONAN_TENLOAI)));

        loaiMonAnDTOS.add(loaiMonAnDTO);

        cursor.moveToNext();
    }

    return loaiMonAnDTOS;

}
```

3.5.3. Thêm món ăn

Đi với việc thêm loại món ăn, chủ nhà hàng thực hiện chức năng thêm món ăn khi đã có loại món ăn.

```
public boolean ThemMonAn(MonAnDTO monAnDTO){
    ContentValues contentValues = new ContentValues();
    contentValues.put(CreateDatabase.TB_MONAN_TENMONAN,monAnDTO.getTenMonAn());
    contentValues.put(CreateDatabase.TB_MONAN_GIATIEN,monAnDTO.getGiaTien());
    contentValues.put(CreateDatabase.TB_MONAN_MALOAI,monAnDTO.getMaLoai());
    contentValues.put(CreateDatabase.TB_MONAN_HINHANH,monAnDTO.getHinhAnh());

    long kiemtra = database.insert(CreateDatabase.TB_MONAN, nullColumnHack: null,contentValues);
    if (kiemtra != 0){
        return true;
    }else {
        return false;
    }
}
```

3.5.4. Lấy danh sách món ăn

Việc lấy danh sách món ăn giúp cho chủ nhà hàng quản lý được trong mỗi loại món ăn gồm có những món ăn nào.

```
public List<MonAnDTO> LayDanhSachMonAnTheoLoai(int maloai){
    List<MonAnDTO> monAnDTOS = new ArrayList<>();

    String truyvan = "SELECT * FROM " + CreateDatabase.TB_MONAN +
            " WHERE " + CreateDatabase.TB_MONAN_MALOAI + " = '" + maloai + "'";
    Cursor cursor = database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        MonAnDTO monAnDTO = new MonAnDTO();

        monAnDTO.setHinhAnh(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_HINHANH)));
        monAnDTO.setTenMonAn(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_TENMONAN)));
        monAnDTO.setGiaTien(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_GIATIEN)));
        monAnDTO.setMaMonAn(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_MONAN_MAMON)));
        monAnDTO.setMaLoai(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_MONAN_MALOAI)));

        monAnDTOS.add(monAnDTO);
        cursor.moveToNext();
    }
    return monAnDTOS;
}
```

3.6. Quản lý thống kê

3.6.1. Lấy danh sách thống kê theo ngày

Việc kiểm tra thống kê theo ngày giúp cho nhà hàng kiểm kê số lượng món ăn được bán ra, doanh thu từng ngày mà nhà hàng hoạt động. Giúp chủ nhà hàng có những so sánh chính xác.

```
public List<ThongKeDTO> LayDanhSachThongKe(String ngaygoi){

    String truyvan = "SELECT " + CreateDatabase.TB_MONAN_TENMONAN + "," + CreateDatabase.TB_CHITIETGOIMON_SOLUONG +
            "," + CreateDatabase.TB_MONAN_GIATIEN + "," + CreateDatabase.TB_GOIMON_NGAYGOI + ","
            + CreateDatabase.TB_MONAN_HINHANH +
            " FROM " + CreateDatabase.TB_CHITIETGOIMON + " ct," + CreateDatabase.TB_GOIMON + " gm,"
            + CreateDatabase.TB_MONAN + " ma WHERE " + "ct." + CreateDatabase.TB_CHITIETGOIMON_MAGOIMON + " = gm."
            + CreateDatabase.TB_GOIMON_MAGOIMON + " AND ct." + CreateDatabase.TB_CHITIETGOIMON_MAMONAN + " = ma."
            + CreateDatabase.TB_MONAN_MAMON + " AND " + CreateDatabase.TB_GOIMON_NGAYGOI + " = '" + ngaygoi + "'";

    List<ThongKeDTO> thongKeDTOS = new ArrayList<ThongKeDTO>();
    Cursor cursor = database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        ThongKeDTO thongKeDTO = new ThongKeDTO();
        thongKeDTO.setNgaygoi(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_GOIMON_NGAYGOI)));
        thongKeDTO.setTenmonan(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_TENMONAN)));
        thongKeDTO.setSoluong(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_CHITIETGOIMON_SOLUONG)));
        thongKeDTO.setGiatien(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_MONAN_GIATIEN)));
        thongKeDTO.setHinhanh(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_HINHANH)));

        thongKeDTO.setThanhtien(thongKeDTO.getSoluong()*thongKeDTO.getGiatien());

        thongKeDTOS.add(thongKeDTO);

        cursor.moveToNext();
    }
    return thongKeDTOS;
}
```

3.6.2. Lấy danh sách thống kê từ ngày đến ngày và thống kê tháng

Việc kiểm tra thống kê theo theo một khoảng thời gian mà nhà hàng hoạt động. Giúp chủ nhà hàng có những sự so sánh chính xác, từ đó đưa ra được những thay đổi, định hướng đúng cho sự phát triển quán nhà hàng.

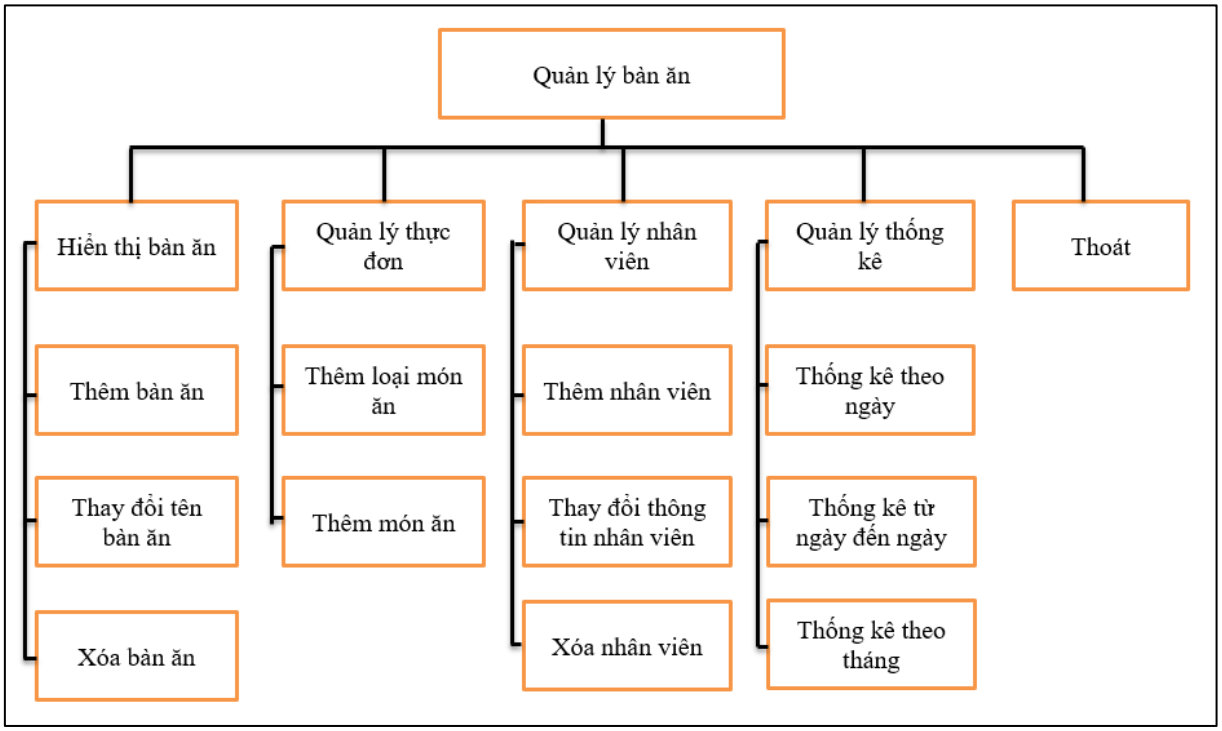
```
public List<ThongKeDTO> LayDanhSachThongKeTuNgayDenNgay(String tungay, String denngay){

    String truyvan = "SELECT " + CreateDatabase.TB_MONAN_TENMONAN + "," + CreateDatabase.TB_CHITIETGOIMON_SOLUONG +
            "," + CreateDatabase.TB_MONAN_GIATIEN + "," + CreateDatabase.TB_GOIMON_NGAYGOI + ","
            + CreateDatabase.TB_MONAN_HINHANH +
            " FROM " + CreateDatabase.TB_CHITIETGOIMON + " ct," + CreateDatabase.TB_GOIMON + " gm,"
            + CreateDatabase.TB_MONAN + " ma WHERE " + "ct." + CreateDatabase.TB_CHITIETGOIMON_MAGOIMON + " = gm."
            + CreateDatabase.TB_GOIMON_MAGOIMON + " AND ct." + CreateDatabase.TB_CHITIETGOIMON_MAMONAN + " = ma."
            + CreateDatabase.TB_MONAN_MAMON + " AND " + CreateDatabase.TB_GOIMON_NGAYGOI + " >= '" + tungay + "' AND "
            +CreateDatabase.TB_GOIMON_NGAYGOI + " <= '" + denngay + "'";

    List<ThongKeDTO> thongKeDTOS = new ArrayList<~>();
    Cursor cursor = database.rawQuery(truyvan, selectionArgs: null);
    cursor.moveToFirst();
    while (!cursor.isAfterLast()){
        ThongKeDTO thongKeDTO = new ThongKeDTO();
        thongKeDTO.setNgaygoi(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_GOIMON_NGAYGOI)));
        thongKeDTO.setTenmonan(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_TENMONAN)));
        thongKeDTO.setSoluong(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_CHITIETGOIMON_SOLUONG)));
        thongKeDTO.setGiatien(cursor.getInt(cursor.getColumnIndex(CreateDatabase.TB_MONAN_GIATIEN)));
        thongKeDTO.setHinhanh(cursor.getString(cursor.getColumnIndex(CreateDatabase.TB_MONAN_HINHANH)));

        thongKeDTO.setThanhtien(thongKeDTO.getSoluong()*thongKeDTO.getGiatien());

        thongKeDTOS.add(thongKeDTO);

        cursor.moveToNext();
    }
    return thongKeDTOS;
}
```

CHƯƠNG 4
CHƯƠNG TRÌNH ỨNG DỤNG THỰC NGHIỆM

4.1. Phát biểu bài toán

Hiện nay, việc quản lý bàn ăn tại các nhà hàng đều được phát triển trên ứng dụng của điện thoại, máy tính bảng. Nhu cầu của khách hàng là phải nhanh, chính xác, nếu vẫn quản lý theo cách thủ công sẽ mất rất nhiều thời gian lẫn công sức mà hiệu quả công việc không được cao, thậm chí sẽ không thể cạnh tranh được với các nhà hàng khác. Vậy nên việc nâng cấp quy trình làm việc, sử dụng công nghệ trong việc quản lý bàn ăn tại nhà hàng cũng như tăng chất lượng phục vụ là điều tất yếu.

4.2. Sơ đồ chức năng



Hình 4.2.1: Sơ đồ chức năng

4.3. Cơ sở dữ liệu

Bảng nhân viên:

STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MANV	Integer		Khóa chính
2	TENDN	Text		
3	MATKHAU	Text		
4	GIOITINH	Text		
5	NGAYSINH	Text		
6	CMND	Text		

Bảng Bàn ăn:

STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MABAN	Integer		Khóa chính
2	TENBAN	Text		
3	TINHTRANG	Text		

Bảng Món ăn:

STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MAMONAN	Integer		Khóa chính
2	TENMONAN	Text		
3	GIATIEN	Integer		
4	MALOAI	Integer		Khóa ngoài
5	HINHANH	Text		

Bảng Loại món ăn:

STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MALOAI	Integer		Khóa chính
2	TENLOAI	Text		

Bảng Gọi món:

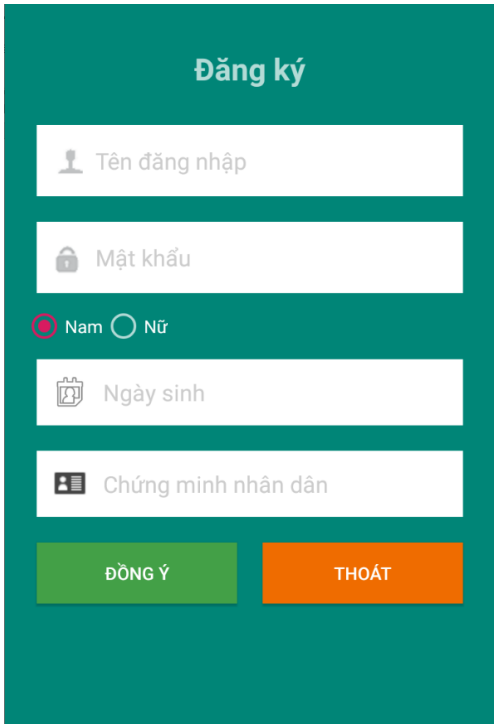
STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MAGOIMON	Integer		Khóa chính
2	MANV	Integer		
3	NGAYGOI	Text		
4	TINHTRANG	Text		
5	MABAN	Integer		Khóa ngoài

Bảng Chi tiết gọi món:

STT	Tên trường	Kiểu trường	Độ rộng	Ghi chú
1	MAGOIMON	Integer		Khóa chính
2	MAMONAN	Integer		Khóa chính
3	SOLUONG	Integer		

4.4. Giao diện chương trình

4.4.1. Giao diện đăng ký

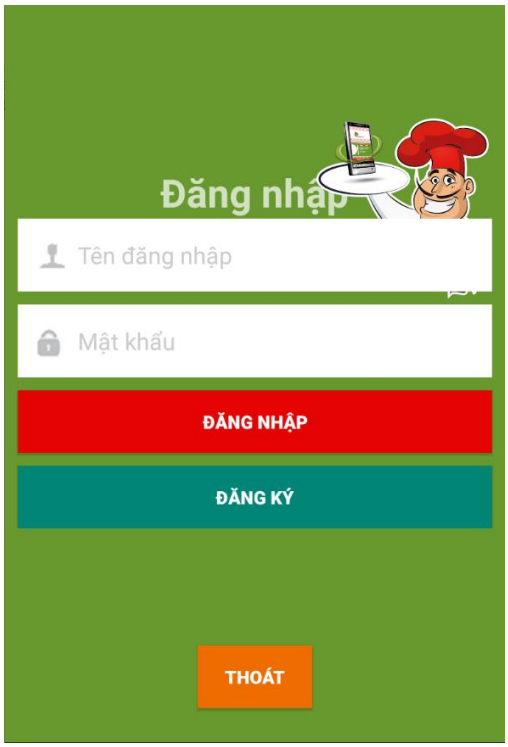


Hình 4.4.1.1: Giao diện đăng ký

Khi mới cài đặt, ứng dụng chưa có dữ liệu của bất kì nhân viên nào, lúc này ta phải thực hiện đăng ký tài khoản.

Nhấn vào nút "ĐĂNG KÝ" ở bên màn hình đăng nhập, sẽ hiện ra giao diện mới cho người đăng ký. Các trường đăng ký bao gồm "Tên đăng nhập", "Mật khẩu", "Giới tính", "Ngày sinh", và "Chứng minh nhân dân". Nếu điền thiếu các trường thông tin thì sẽ không hợp lệ. Nếu hoàn thành xong form đăng ký này chính xác và nhấn nút "ĐỒNG Ý" thì chúng ta đã đăng ký thành công, sau đó nhấn nút thoát, quay lại giao diện đăng nhập.

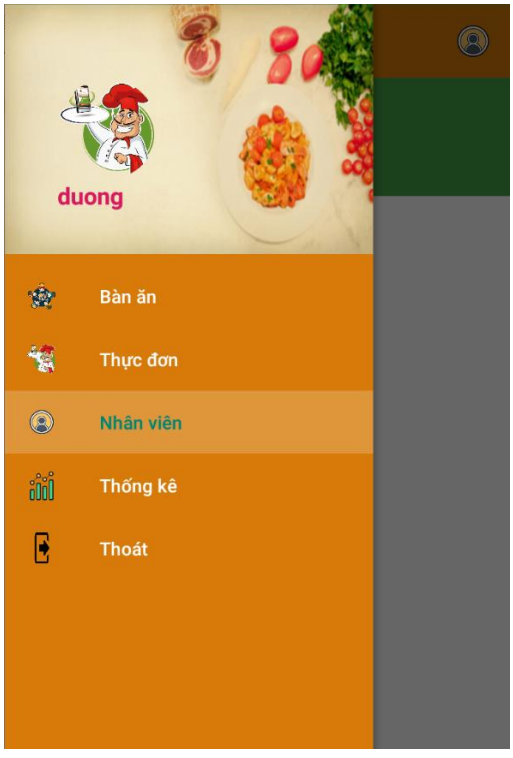
4.4.2. Giao diện đăng nhập



Hình 4.4.2.1: Màn hình đăng nhập

Với mục đích bảo mật thì chúng ta sẽ có màn hình để đăng nhập vào ứng dụng, đặc biệt tránh trường hợp bạn cho người khác mượn điện thoại hay máy tính bảng của mình. Khi mới cài đặt ứng dụng, bạn chưa đăng ký tài khoản và phải nhấn vào nút "ĐĂNG KÝ" để đăng ký tài khoản mới. Nếu đã đăng ký tài khoản rồi thì điền đầy đủ thông tin chính xác và nhấn vào nút "ĐĂNG NHẬP" là đăng nhập thành công, ngược lại sai thông tin sẽ không đăng nhập thành công.

4.4.3. Giao diện Menu

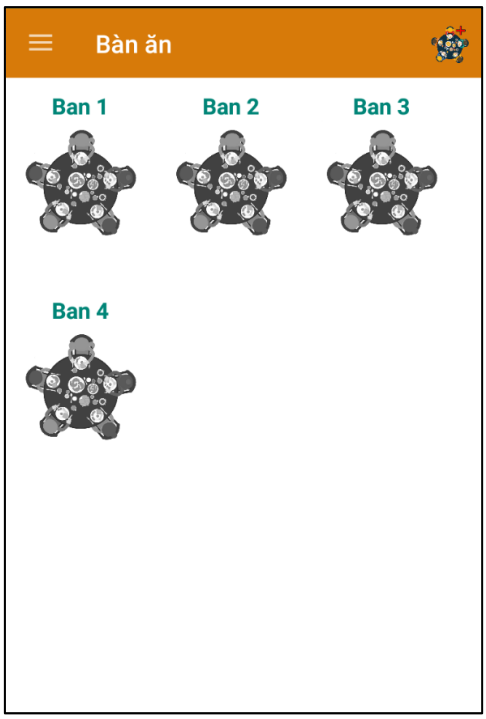


Hình 4.4.3.1: Giao diện Menu

Chúng ta có thể truy cập vào các giao diện khác bằng cách nhấn vào biểu tượng 3 gạch ở góc trên bên trái màn hình. Ở đây sẽ xuất hiện ra tất cả chức năng của ứng dụng cho chúng ta chọn: đó là Bàn ăn, Thực đơn, Nhân viên, Thống kê.

Lưu ý, ở phía dưới hình ảnh đầu bếp có tên đăng nhập của nhân viên.

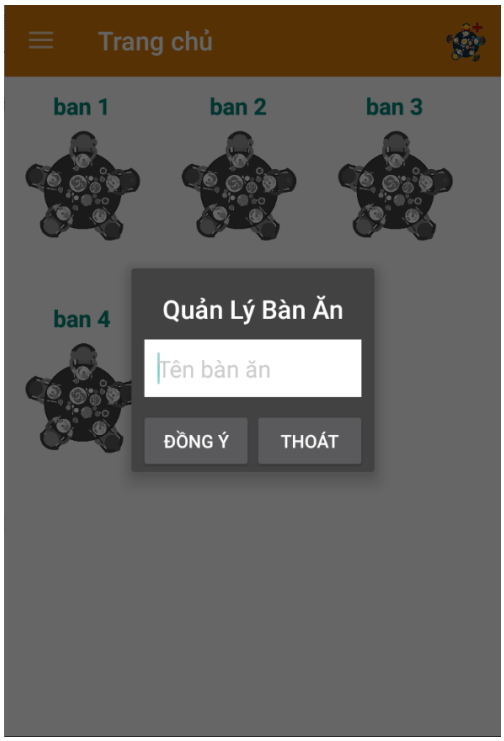
4.4.4. Giao diện bàn ăn



Giả sử trong nhà hàng có 4 bàn ăn, nên ta thêm 4 bàn ăn vào giao diện trang chủ như chúng ta thấy ở hình ảnh 4 bàn ăn. Chúng ta thêm bàn ăn cho nhà hàng bằng cách nhấn vào hình ảnh bàn ăn có dấu cộng ở góc trên bên phải màn hình sẽ xuất hiện một popup cho chúng ta thêm bàn ăn.

Hình 4.4.4.1: Giao diện bàn ăn

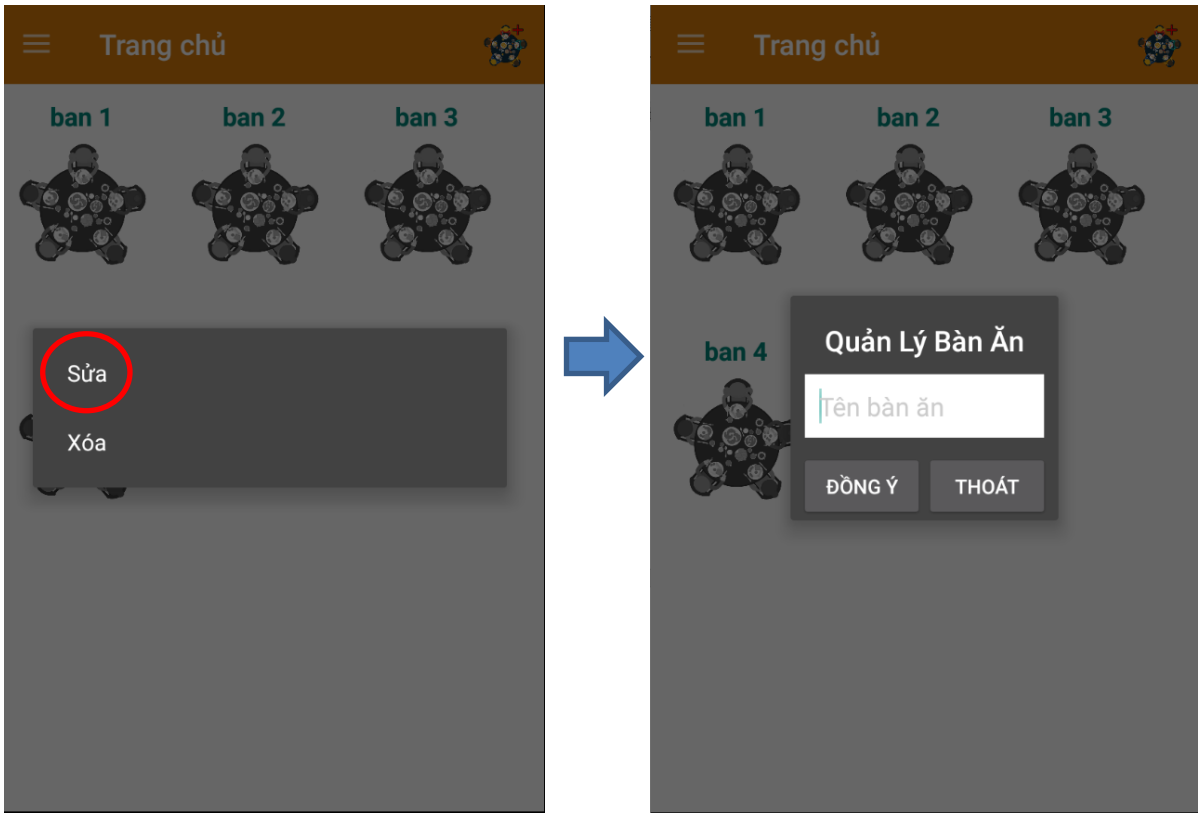
4.4.5. Giao diện thêm bàn ăn



Chúng ta đặt tên bàn mới vào, nếu không có tên sẽ không thêm được bàn ăn. Sau khi thêm tên bàn ăn mới và nhấn vào nút "đồng ý" bàn ăn sẽ cập nhật vào giao diện trang chủ. Và nút "thoát" nếu như ta không thêm bàn ăn mới nữa để trở về giao diện trang chủ.

Hình 4.4.5.1: Giao diện thêm bàn ăn

4.4.6. Giao diện sửa tên bàn ăn

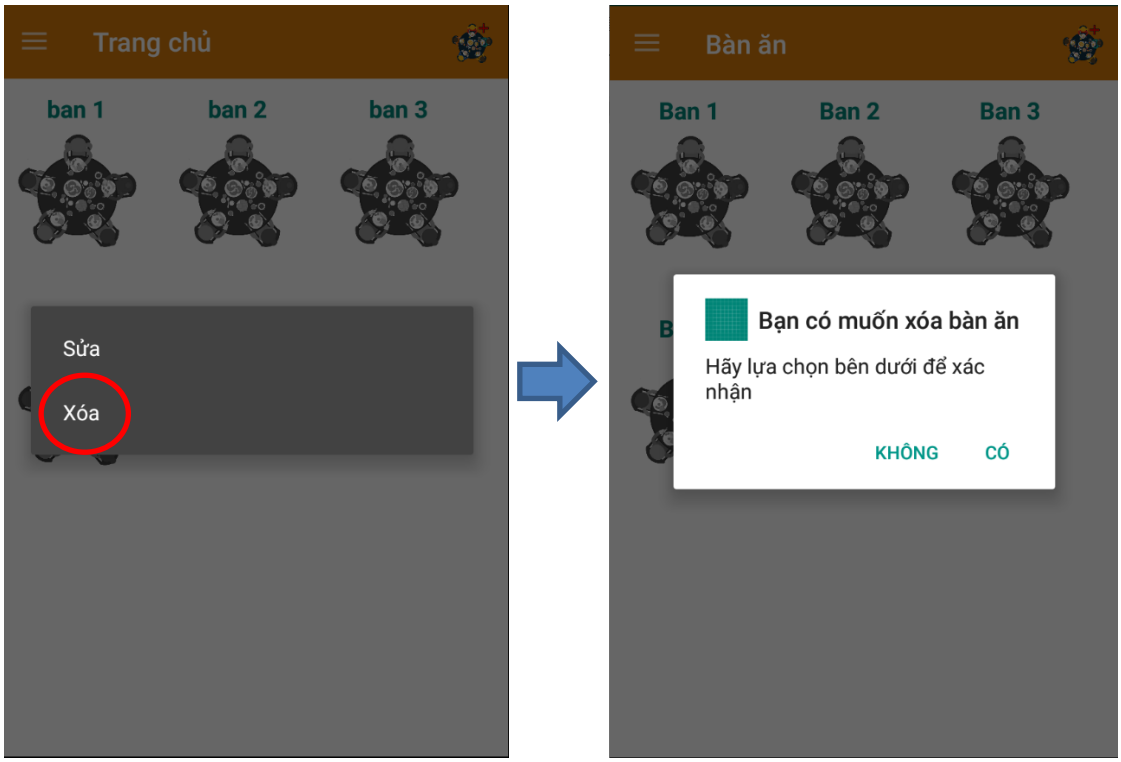


Hình 4.4.6.1: Giao diện sửa tên bàn ăn

Chúng ta có thể sửa tên bàn ăn bằng cách nhấn giữ vào tên bàn sẽ xuất hiện popup như hình. Nhấn "sửa" sẽ xuất hiện một giao diện sửa màn hình cho phép chúng ta sửa tên bàn ăn.

Sau khi nhấn vào "sửa" bàn ăn, sẽ xuất hiện giao diện popup sửa tên bàn ăn (giống với giao diện popup thêm bàn ăn). Cho phép chúng ta sửa tên bàn ăn đã có sẵn. Nhấn nút "đồng ý" sau khi thay tên bàn, hoặc nhấn "thoát" nếu như chúng ta không thay đổi.

4.4.7. Giao diện xóa bàn ăn



Hình 4.4.7.1: Giao diện xóa bàn ăn

Chúng ta có thể xóa bàn ăn bằng cách nhấn giữ vào tên bàn sẽ xuất hiện popup như hình. Nhấn "xóa" sẽ xuất hiện một giao diện màn hình cho phép chúng ta lựa chọn có nên xóa bàn ăn hay không. Nếu "có" thì băn ăn được chọn sẽ bị xóa lập tức, còn "không" nếu như chúng ta không xóa bàn ăn.

4.4.8. Giao diện loại món ăn

Hình 4.4.8.1: Giao diện loại món ăn

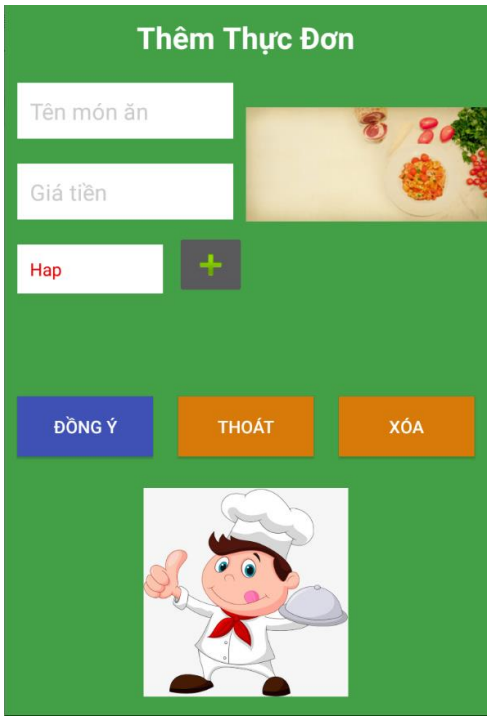
Sau khi nhấn vào "Thực đơn" ở giao diện Menu, giao diện "Loại món ăn" sẽ hiện ra. Ta có thể thấy các loại món ăn trong thực đơn.

4.4.9. Giao diện món ăn

Hình 4.4.9.1: Giao diện món ăn

Khi nhấn vào hình ảnh của từng loại món ăn, ta sẽ thấy tất cả các món ăn có trong từng loại món ăn. Ví dụ ở đây là nhấn vào hình ảnh loại món ăn "Hap".

4.4.10. Giao diện thêm món ăn



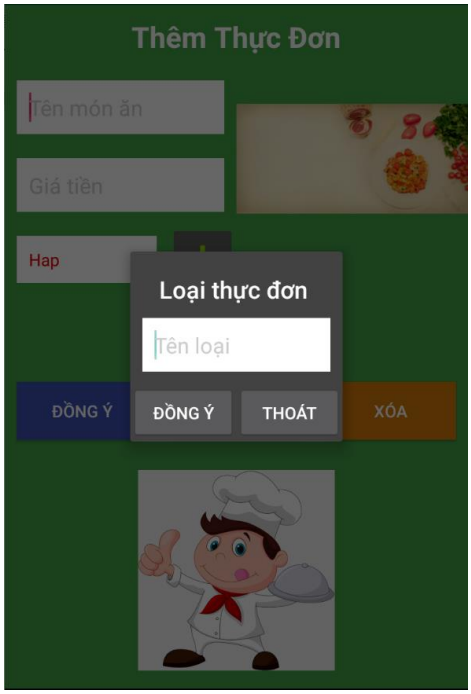
Hình 4.4.10.1: Giao diện thêm thực đơn

Ta muốn thêm món ăn và loại món ăn bằng cách trở về giao diện "Loại thực đơn" và nhấn vào hình ảnh góc trên bên phải màn hình. Sẽ xuất hiện ra giao diện "Thêm thực đơn".

Ta bắt đầu thêm tên món ăn, giá tiền, thêm ảnh món ăn, sau đó chọn loại món ăn bằng cách nhấn vào ô bên trái nút dấu cộng. Sẽ xuất ra những loại món ăn đã có để chúng ta chọn.

Sau khi điền đủ thông tin yêu cầu, nhấn "đồng ý" để thêm món ăn, như vậy ta đã thêm thành công món ăn ứng với loại món ăn đó. Nhấn "xóa" nếu cần thêm món ăn mới nữa hoặc nhấn "thoát" để xem loại món ăn và món ăn đã được cập nhật.

4.4.11. Giao diện thêm loại món ăn



Hình 4.4.11.1: Giao diện thêm loại món ăn

Nếu món ăn của chúng ta cần trong loại món ăn mới nhưng chưa có, chúng ta nhấn vào nút dấu cộng để thêm loại món ăn.

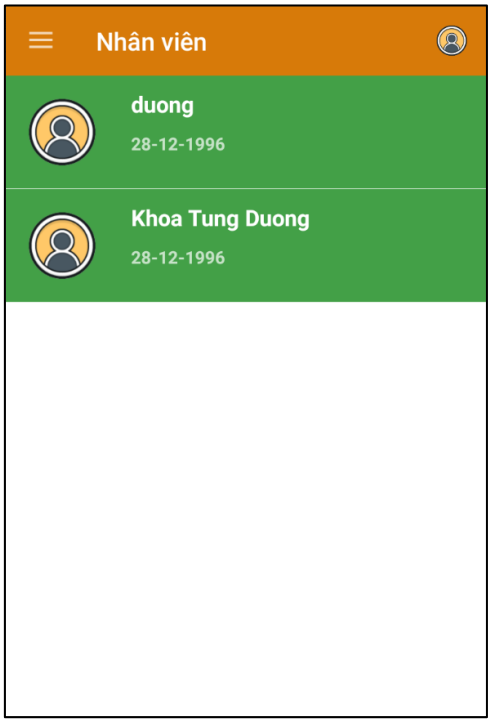
Nhập tên loại món ăn mới vào (nếu không nhập sẽ không thể thêm), nhấn "thoát" nếu không thêm loại món ăn hoặc nhấn "đồng ý" nếu thêm loại món ăn, sau đó sẽ trở lại giao diện "thêm thực đơn" .

4.4.12. Giao diện chọn ảnh món ăn

Để thêm hình ảnh món ăn, ta nhấn vào ô hình ảnh trong giao diện "thêm thực đơn", sẽ xuất hiện ra tất cả các hình ảnh món ăn mà ta cần thêm.

Hình 4.4.12.1: Chọn ảnh món ăn

4.4.13. Giao diện danh sách nhân viên

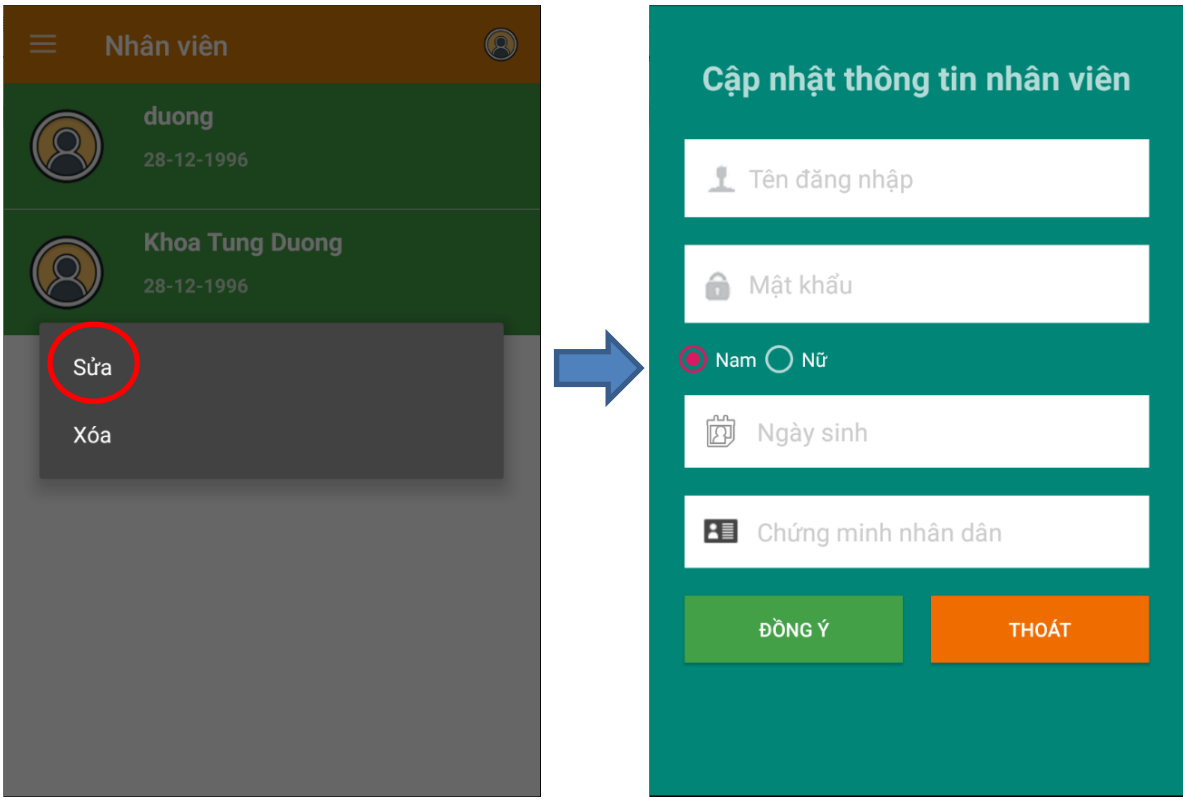


Chọn giao diện "nhân viên" ở Menu, ở đây chúng ta có thể thấy được tất cả các nhân viên.

Nhấn vào biểu tượng nhân viên ở góc trên bên phải màn hình, sẽ xuất hiện giao diện đăng ký.

Hình 4.4.13.1: Giao diện nhân viên

4.4.14. Giao diện cập nhật thông tin nhân viên

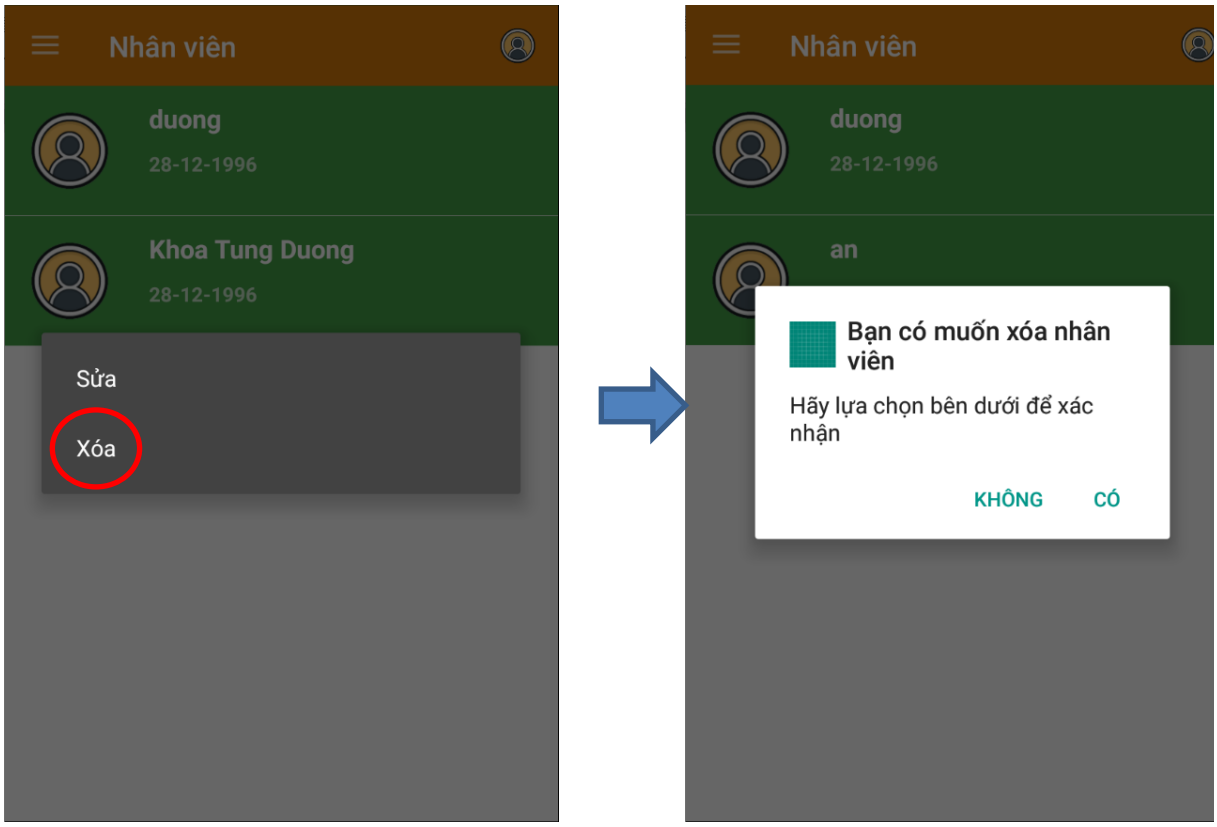


Hình 4.4.14.1: Giao diện cập nhật thông tin nhân viên

Chúng ta có thể sửa thông tin nhân viên bằng cách nhấn giữ vào nhân viên đó, sẽ xuất hiện popup xóa hoặc sửa cho chúng ta lựa chọn.

Chọn "sửa" sẽ xuất hiện giao diện cập nhật thông tin nhân, nhập dữ liệu và nhấn "đồng ý" khi muốn cập nhật thông tin nhân viên, hoặc "thoát" khi không thay đổi.

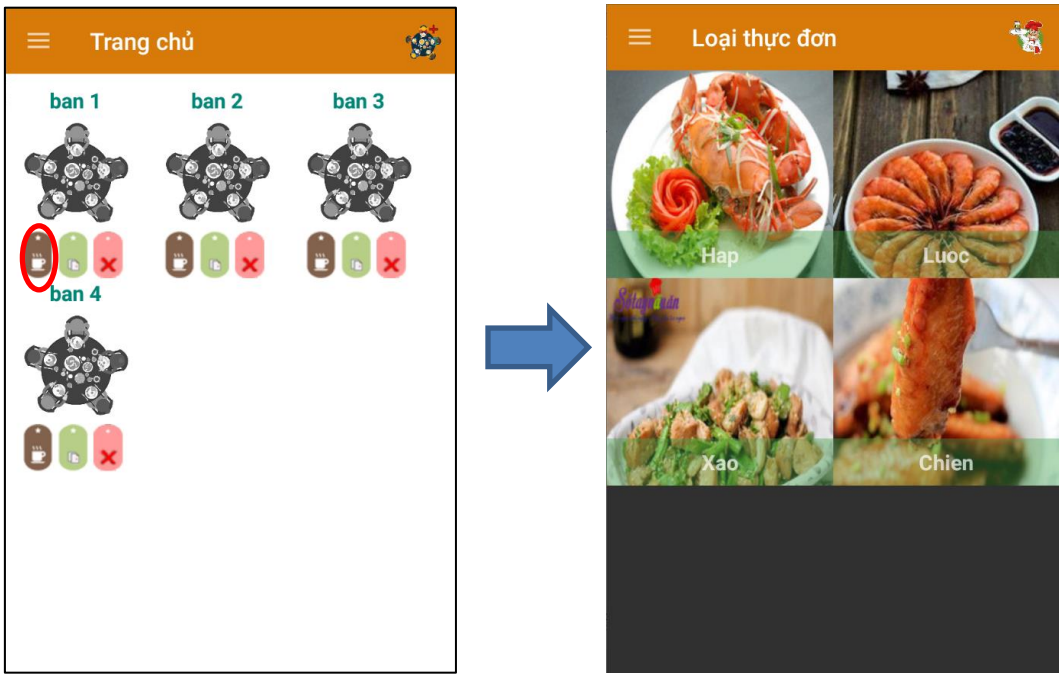
4.4.15. Giao diện xóa nhân viên

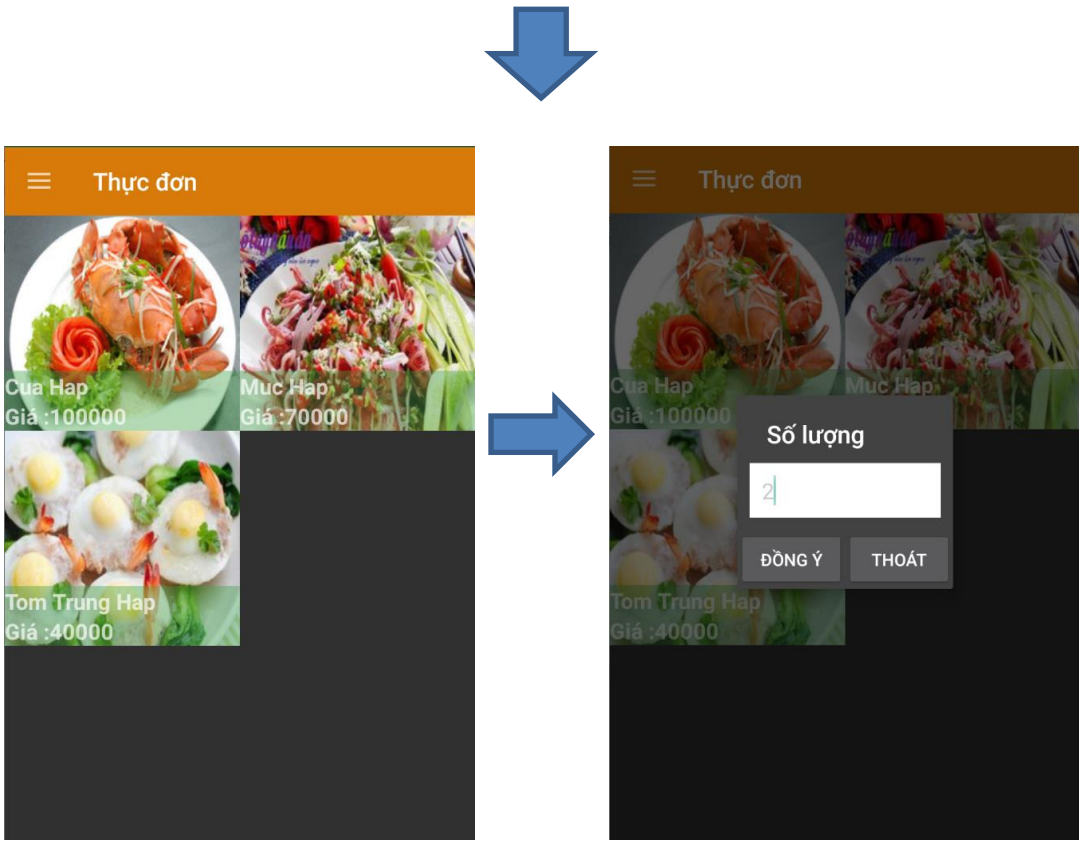


Hình 4.4.15.1: Giao diện xóa nhân viên

Chọn popup "xóa" sẽ chuyển đến giao diện thông báo xóa nhân viên. Ở đây, chúng ta chọn "có" nếu muốn xóa nhân viên, hoặc chọn "không" nếu như không có gì thay đổi.

4.4.16. Giao diện gọi món

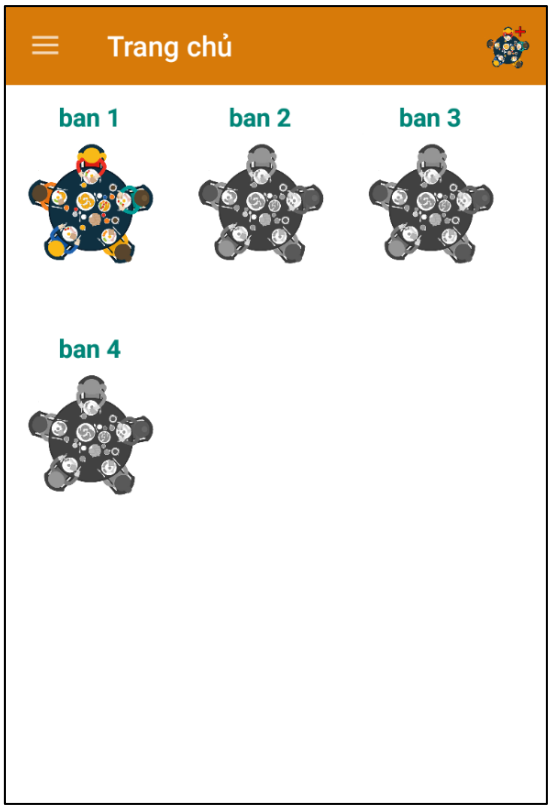




Hình 4.4.16.1: Giao diện gọi món

Ở giao diện bàn ăn, nhấn vào nút "gọi món" màu nâu để bắt đầu gọi món, sau đó chọn loại món ăn, món ăn và thêm số lượng món ăn cần gọi.

4.4.17. Giao diện bàn ăn đổi màu sau khi gọi món ăn

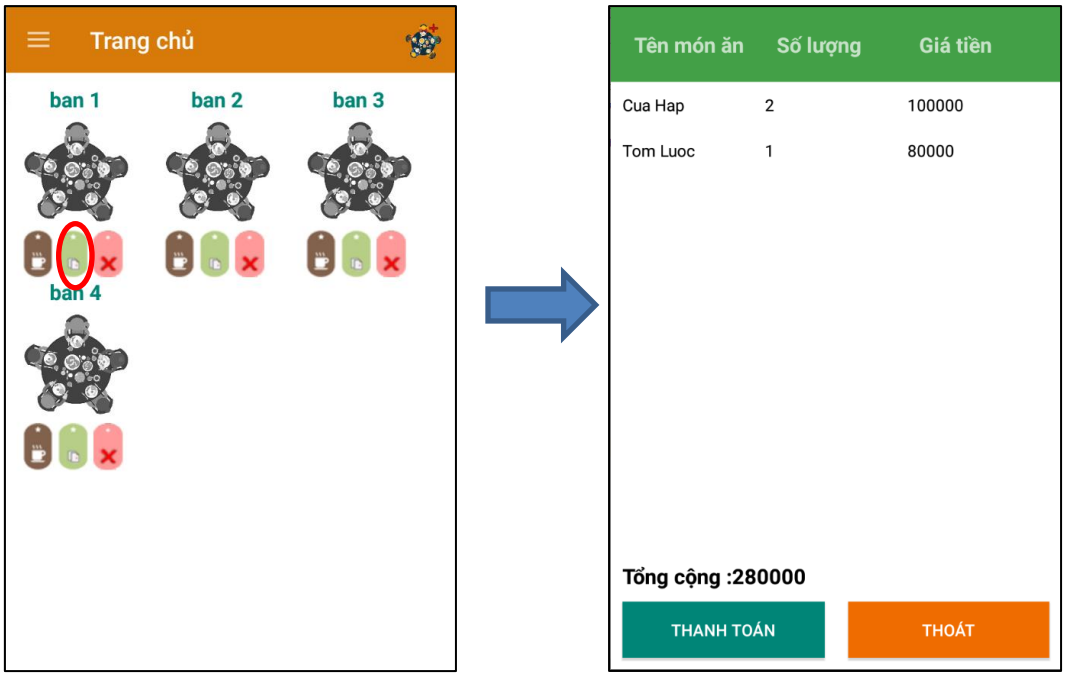


Hình 4.4.17.1: Giao diện bàn ăn đổi màu sau khi gọi món

Gọi món ăn thành công, khi trở lại trang chủ "ban 1" của chúng ta ngay lập tức có màu chứng tỏ bàn ăn đó đã có người ngồi và có gọi món ăn.

Có thể thêm món ăn khác vào bàn hoặc cùng 1 món thêm số lượng tùy thích, bàn ăn đó gọi thêm thì sẽ cập nhật chính xác số lượng món ăn đã gọi.

4.4.18. Giao diện Thanh toán



Hình 4.4.18.1: Giao diện thanh toán

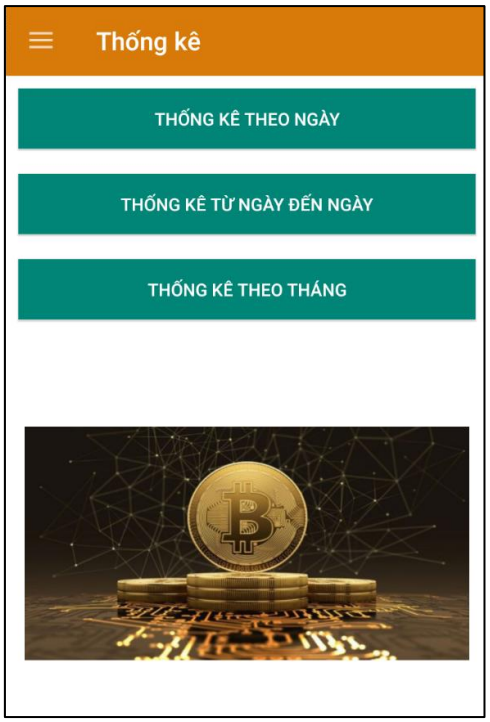
Khi khách hàng muốn thanh toán, người dùng chỉ việc nhấn vào hình ảnh nút "thanh toán" màu xanh ở giao diện trang chủ và giao diện thanh toán xuất hiện.

Ở giao diện này đã liệt kê chính xác tên món ăn, số lượng, giá tiền của từng món và tổng tiền. Chúng ta chỉ việc nhấn nút "thanh toán".

Sau khi nhấn nút thanh toán các dữ liệu sẽ mất đi và nhấn nút thoát để trở về giao diện trang chủ.

Lúc này bàn ăn vừa được thanh toán xong đã chuyển sang màu xám chứng tỏ bàn ăn vừa rồi đã trống (không có khách ngồi).

4.4.19. Giao diện thống kê



Hình 4.4.19.1: Giao diện thống kê

Chọn giao diện "thống kê" ở Menu.

Xuất hiện giao diện thống kê gồm có các mục: Thống kê theo ngày, thống kê từ ngày đến ngày và thống kê theo tháng.

4.4.20. Giao diện thống kê theo ngày

Hình 4.4.20.1: Giao diện thống kê theo ngày

Giao diện thống kê nhấn nút "Thống kê theo ngày" sẽ xuất hiện giao diện "Doanh thu theo ngày".

Ở đây có 1 ô chọn ngày cần thống kê.

Như ở hình, nhấn nút "thống kê" sẽ hiện ra danh sách các món ăn đã gọi trong ngày 2019-09-18.

Khi thống kê xong nhấn "thoát" để trở về giao diện thống kê.

4.4.21. Giao diện thống kê từ ngày đến ngày

Hình 4.4.21.1: Giao diện Thống kê từ ngày đến ngày

Giao diện thống kê nhấn nút "Thống kê từ ngày đến ngày", sẽ xuất hiện giao diện "Doanh thu từ ngày đến ngày".

Ở đây có 2 ô để nhập ngày.

Như ở hình, ô đầu tiên nhập từ ngày 2019-08-29, ô nhập thứ hai là đến ngày 2019-09-18. Nhấn nút "thống kê" sẽ hiện ra danh sách các món ăn được gọi trong khoảng thời gian đã chọn.

Khi thống kê xong nhấn "thoát" để trở về giao diện thống kê.

4.4.22. Giao diện thống kê theo tháng

Hình 4.4.22.1:Giao diện thống kê tháng

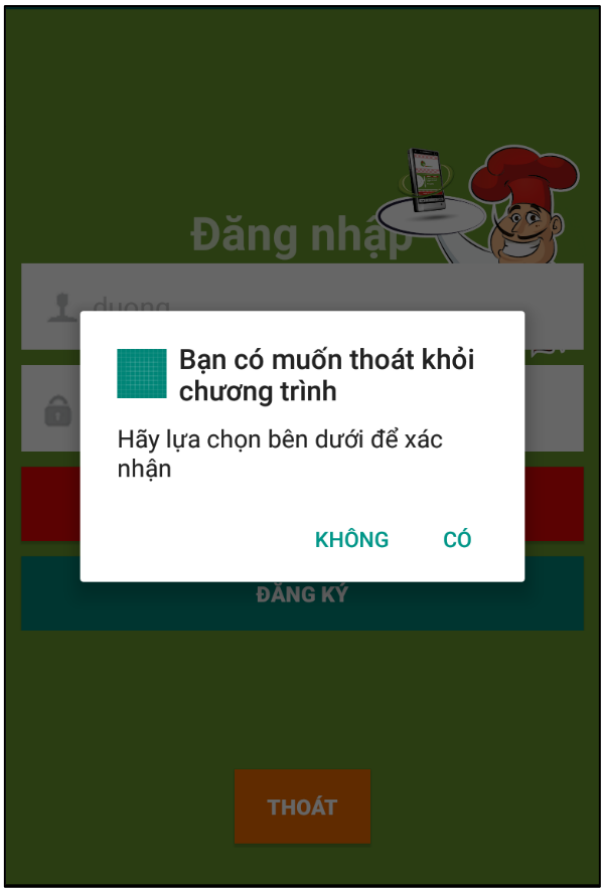
Giao diện thống kê nhấn nút "Thống kê theo tháng", sẽ xuất hiện giao diện "Doanh thu theo tháng".

Ở đây có 2 ô để nhập ngày.

Như ở hình, ô đầu tiên nhập ngày đầu tiên của tháng cần thống kê, ô nhập thứ hai là ngày cuối cùng của tháng cần thống kê. Nhấn nút "thống kê" sẽ hiện ra danh sách các món ăn được gọi trong tháng đã chọn.

Khi thống kê xong nhấn "thoát" để trở về giao diện thống kê.

4.4.23. Giao diện thoát ứng dụng



Hình 4.4.23.1: Giao diện thoát ứng dụng

Ở giao diện đăng nhập, khi nhấn nút "thoát" sẽ xuất hiện một khung thông báo thoát khỏi chương trình.

Nhấn nút "có" nếu muốn thoát ứng dụng.

Nhấn nút "không" nếu không muốn thoát khỏi ứng dụng.

KẾT LUẬN

Trên đây em đã khảo sát trên mặt lý thuyết đối với xây dựng ứng dụng Android. Đồ án hướng tới mục tiêu xây dựng ứng dụng quản lý bàn ăn trên thiết bị di động Android. Trong khoảng thời gian nhất định dành cho việc thực hiện đề tài, nên một số vấn đề vẫn chưa được hoàn chỉnh. Tuy nhiên, đồ án đã đạt được một số kết quả:

Kết quả đạt được:

- Nắm được một số kỹ thuật hay sử dụng và cách sử dụng của lập trình Android.

- Hoàn thành các chức năng của ứng dụng quản lý bàn ăn như: quản lý bàn ăn, quản lý nhân viên, quản lý thực đơn, quản lý thống kê.

Hạn chế:

Trong thời gian qua, em đã cố gắng để tìm hiểu thực hiện đề tài. Tuy nhiên với kinh nghiệm và thời gian hạn chế nên không thể tránh khỏi những thiếu sót trong đồ án. Cụ thể:

- Chưa nghiên cứu sâu vào các kỹ thuật của lập trình Android.

- Trình bày thiếu logic, cách diễn đạt còn kém.

Trong thời gian tới, em sẽ tiếp tục tìm hiểu và nghiên cứu cũng như hoàn thiện đồ án của mình ở mức cao nhất.

DANH MỤC TÀI LIỆU THAM KHẢO

Tài liệu tham khảo online:

[1]:https://vi.wikipedia.org/wiki/Android_(h%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh)

[2]: https://xuanthulab.net/android-java/