

**BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN
TRƯỜNG ĐẠI HỌC THỦY LỢI**



BÀI TẬP LỚN

**HỌC PHẦN: PHÁT TRIỂN ỨNG DỤNG
CHO THIẾT BỊ DI ĐỘNG**

**ĐỀ TÀI: Xây dựng ứng dụng bán đồ ăn nhanh ChickenGang
(Java Console App)**

Mã Sinh Viên	Họ và Tên	Ngày Sinh	Lớp
2151170551	Nguyễn Duy Tài	18/02/2003	63KTPM2
2151170560	Lê Văn Bình	24/08/2003	63KTPM2
2151173810	Dương Thị Hồng Nhung	25/07/2003	63KTPM2

Hà Nội, năm 2024

BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN
TRƯỜNG ĐẠI HỌC THỦY LỢI



BÀI TẬP LỚN

HỌC PHẦN: PHÁT TRIỂN ỨNG DỤNG
CHO THIẾT BỊ DI ĐỘNG

ĐỀ TÀI: Xây dựng ứng dụng ChickenGang (Java Console App)

Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
			Bảng Số	Bảng Chữ
2151170551	Nguyễn Duy Tài	18/02/2003		
2151170560	Lê Văn Bình	24/08/2003		
2151173810	Dương Thị Hồng Nhung	25/07/2003		

CÁN BỘ CHẤM THI 1

CÁN BỘ CHẤM THI 2

Hà Nội, năm 2024

LỜI NÓI ĐẦU

Hiện nay, việc áp dụng các giải pháp công nghệ vào hoạt động kinh doanh đã trở thành xu hướng tất yếu để nâng cao hiệu quả và cạnh tranh trên thị trường. Đặc biệt, các ứng dụng di động ngày càng trở nên phổ biến và là cầu nối quan trọng giữa doanh nghiệp và khách hàng.

Dự án bài tập lớn được xây dựng với mục tiêu phát triển một ứng dụng di động dành cho cửa hàng bán đồ ăn nhanh ChickenGang. Ứng dụng không chỉ giúp khách hàng dễ dàng tiếp cận các món ăn của cửa hàng mà còn mang đến những tiện ích thiết thực như đăng nhập, đăng ký tài khoản, chỉnh sửa thông tin cá nhân, đặt bàn, đặt món mang về, tìm kiếm món ăn, và xem các địa điểm cửa hàng. Với việc tích hợp dữ liệu từ máy chủ và lưu trữ cục bộ, ứng dụng đảm bảo cung cấp thông tin chính xác và kịp thời cho người dùng.

Dự án này không chỉ giúp chúng em hiểu rõ hơn về quy trình phát triển ứng dụng Android mà còn rèn luyện các kỹ năng như quản lý dữ liệu, xử lý giao diện người dùng, và tích hợp các dịch vụ web. Qua đó, có cơ hội áp dụng những kiến thức đã học vào thực tiễn, nâng cao khả năng lập trình cũng như làm việc nhóm.

Mặc dù đã cố gắng hết sức, báo cáo vẫn không tránh khỏi những thiếu sót. Nhóm mong nhận được những góp ý từ giáo viên và các bạn để có thể hoàn thiện hơn về kiến thức và kỹ năng.

MỤC LỤC

LỜI NÓI ĐẦU	3
MỤC LỤC	4
MỤC LỤC HÌNH ẢNH	6
MỤC LỤC BẢNG	7
BẢNG CÁC TỪ VIẾT TẮT	8
PHÂN CHIA CÔNG VIỆC	9
CHƯƠNG 1. MÔ TẢ BÀI TOÁN	10
1.1. Giới thiệu	10
1.2. Chức năng chính	10
1.3. Yêu cầu phi chức năng	11
CHƯƠNG 2. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG	12
2.1. Phân tích yêu cầu:	12
2.2. Thiết kế hệ thống:	15
2.2.1 Biểu đồ lớp cho mô hình miền (ER Diagram):	15
2.2.2 Biểu đồ lớp (Class Diagram):	15
2.3 Thiết kế cơ sở dữ liệu:	21
2.4 Thiết kế giao diện:	22
CHƯƠNG 3. KẾT QUẢ THỰC HIỆN	33
3.1. Công nghệ đã sử dụng	33
3.2. Tiến độ thực hiện	33
3.3. Hình ảnh sản phẩm	39

MỤC LỤC HÌNH ẢNH

[Hình 1 : Biểu đồ lớp cho mô hình miền \(ER Diagram\)](#)

[Hình 2: Biểu đồ lớp \(Class Diagram\)](#)

[Hình 3: Cấu trúc database](#)

[Hình 4: Giao diện đăng nhập](#)

[Hình 5: Giao diện đăng ký](#)

[Hình 6: Giao diện xem thông tin cá nhân](#)

[Hình 7: Giao diện chỉnh sửa thông tin cá nhân](#)

[Hình 8: Giao diện đăng xuất](#)

[Hình 9: Giao diện trang chủ](#)

[Hình 10: Giao diện danh mục](#)

[Hình 11: Giao diện chi tiết sản phẩm](#)

[Hình 12: Giao diện giỏ hàng](#)

[Hình 13: Giao diện chọn mã giảm giá](#)

[Hình 14: Giao diện xác nhận thông tin thanh toán](#)

[Hình 15: Giao diện chọn vị trí nhận hàng](#)

[Hình 18: Giao diện theo dõi đơn hàng](#)

[Hình 19 : Giao diện đặt bàn](#)

[Hình 20: Giao diện chi tiết đặt bàn](#)

[Hình 21: Giao diện bàn đã đặt](#)

[Hình 22: Giao diện chi tiết bàn đã đặt](#)

[Hình 23: Đăng nhập](#)

[Hình 24: Đăng ký](#)

[Hình 25: Trang chủ](#)

[Hình 26: Danh mục](#)

[Hình 27: Chi tiết sản phẩm](#)

[Hình 28: Đặt bàn](#)

[Hình 29: Bàn đã đặt](#)

[Hình 30: Chi tiết bàn đã đặt](#)

[Hình 31: Giỏ hàng](#)

[Hình 32: Xem thông tin cá nhân](#)

[Hình 33: Chỉnh sửa thông tin cá nhân](#)

[Hình 34: Đăng xuất](#)

MỤC LỤC BẢNG

Bảng 1: Phân chia công việc

BẢNG CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	CSDL	Cơ sở dữ liệu
2	SĐT	Số điện thoại

PHÂN CHIA CÔNG VIỆC

Họ và tên	Công việc	Ghi chú
Dương Thị Hồng Nhung	<p>Giao diện và chức năng:</p> <ul style="list-style-type: none"> • Đăng ký • Đăng nhập • Đăng xuất • Xem thông tin cá nhân • Chỉnh sửa thông tin cá nhân 	
Lê Văn Bình	<p>Giao diện và chức năng:</p> <ul style="list-style-type: none"> • Trang chủ • Danh mục • Tìm kiếm món ăn • Giỏ hàng • Đơn hàng • Thanh toán 	
Nguyễn Duy Tài	<p>Giao diện và chức năng:</p> <ul style="list-style-type: none"> • Đặt bàn • Xem bàn đã đặt • Tìm kiếm cơ sở <p>Thiết kế hệ thống (vẽ biểu đồ ER, biểu đồ lớp)</p>	

Bảng 1: Phân chia công việc

CHƯƠNG 1. MÔ TẢ BÀI TOÁN

1.1. Giới thiệu

Chicken Gang là một ứng dụng đặt hàng, đặt bàn và quản lý dịch vụ ăn uống, giúp người dùng dễ dàng tìm kiếm các món ăn, đặt bàn, và theo dõi đơn hàng tại cửa hàng đồ ăn nhanh yêu thích. Ứng dụng được thiết kế với một cơ sở dữ liệu mạnh mẽ, tổ chức thông tin thành nhiều bảng dữ liệu để tối ưu hóa việc quản lý và phục vụ khách hàng.

Với ứng dụng **Chicken Gang**, người dùng có thể thoải mái đặt món, quản lý các đơn hàng và đặt bàn một cách dễ dàng và tiện lợi. Ứng dụng mang đến cho người dùng một trải nghiệm trực tuyến toàn diện, từ việc chọn món đến việc giao hàng, tất cả đều được thực hiện chỉ với vài thao tác đơn giản.

1.2. Chức năng chính

Ứng dụng **Chicken Gang** cần có các chức năng sau:

- Đăng ký: Cho phép người dùng mới tạo tài khoản trên ứng dụng
- Đăng nhập: Cho phép người dùng đăng nhập vào ứng dụng bằng số điện thoại và mật khẩu
- Đăng xuất: Cho phép người dùng thoát khỏi tài khoản của mình khi không sử dụng ứng dụng nữa.
- Xem thông tin cá nhân: Cho phép người dùng xem thông tin cá nhân của mình bao gồm họ tên, số điện thoại, địa chỉ
- Chỉnh sửa thông tin: Cho phép người dùng thay đổi thông tin cá nhân gồm họ tên, địa chỉ, mật khẩu
- Xem chi tiết món ăn:
 - Hiện thị danh sách các món ăn có sẵn tại cửa hàng bao gồm tên, mô tả, giá cả, hình ảnh, và phân loại.
 - Cho phép người dùng lọc món ăn theo phân loại (ví dụ: món chính, món phụ, đồ uống) hoặc tìm kiếm tên món.
- Thêm đơn hàng mới:
 - Cho phép người dùng chọn cửa hàng và thêm các món ăn vào đơn hàng của mình.
- Đặt hàng: Cho phép người dùng mua và thanh toán các đơn hàng của mình

- Đặt chỗ:
 - Cho phép người dùng đặt bàn trước tại cửa hàng, với các thông tin về ngày, giờ, số lượng khách và cửa hàng mong muốn.
 - Có thể xem danh sách các đặt chỗ đã thực hiện và lọc theo ngày đặt hoặc cửa hàng

1.3. Yêu cầu phi chức năng

- Dễ sử dụng: Giao diện cần rõ ràng, dễ hiểu và dễ sử dụng.
- Đẹp mắt : Giao diện ứng dụng cần thu hút người dùng bởi màu sắc, cách bố trí nội dung
- Hiệu năng: Ứng dụng cần hoạt động nhanh chóng và hiệu quả, ngay cả khi có nhiều công việc.
- Độ tin cậy: Dữ liệu công việc cần được lưu trữ an toàn và không bị mất mát.

CHƯƠNG 2. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG

2.1. Phân tích yêu cầu:

Xác định người dùng:

- Người dùng cuối là các khách hàng, người sẽ sử dụng ứng dụng để đặt các món ăn hoặc đặt bàn trước từ quán **Chicken Gang**.
- Người dùng có thể có các mức độ am hiểu về máy tính khác nhau, do đó ứng dụng cần dễ sử dụng và thân thiện với người dùng.

Thu thập yêu cầu:

- Dựa trên mô tả bài toán, ta đã xác định được các chức năng chính (đăng ký, đăng nhập, đăng xuất, xem thông tin cá nhân, chỉnh sửa thông tin cá nhân, xem chi tiết món ăn, thêm đơn hàng mới, đặt hàng, đặt chỗ) và yêu cầu phi chức năng (dễ sử dụng, hiệu năng, độ tin cậy, giao diện đẹp mắt) của ứng dụng **ChickenGang**.

Phân tích yêu cầu:

- **Đăng ký:**
 - Người dùng cần nhập số điện thoại (bắt buộc, phải là duy nhất), mật khẩu (bắt buộc), tên và địa chỉ (tùy chọn).
 - Sau khi đăng ký thành công, mỗi tài khoản sẽ có một ID duy nhất để phân biệt.
- **Đăng nhập:**
 - Người dùng đăng nhập vào ứng dụng bằng cách sử dụng số điện thoại và mật khẩu.
 - Ứng dụng cần xác thực thông tin để đảm bảo tính an toàn và quyền truy cập.
- **Đăng xuất:**
 - Người dùng có thể đăng xuất khỏi tài khoản để đảm bảo tính bảo mật và ngăn chặn truy cập trái phép.
- **Chỉnh sửa thông tin:**
 - Người dùng có thể cập nhật các thông tin cá nhân bao gồm tên, địa chỉ, và mật khẩu.
 - Mỗi thông tin sẽ được gán ID duy nhất để người dùng dễ dàng truy xuất và chỉnh sửa.

- **Xem chi tiết món ăn:**

- Hiện thị danh sách các món ăn sẵn có bao gồm tên, mô tả, giá cả, hình ảnh.
- Cho phép người dùng lọc danh sách món ăn theo phân loại (Gà rán , Đồ ăn vặt, Đồ uống,...) hoặc tìm kiếm theo tên.
- Cần hiển thị các thông tin quan trọng của món ăn: tên món, giá, hình ảnh.

- **Thêm món ăn vào giỏ hàng:**

- Người dùng được phép thêm món ăn vào giỏ hàng và thanh toán sau
- Mỗi người dùng sẽ có một giỏ hàng duy nhất để theo dõi và quản lý

- **Xem chi tiết giỏ hàng:**

- Người dùng được phép xem các sản phẩm đang có trong giỏ hàng của mình
- Người dùng được phép thay đổi số lượng sản phẩm trong giỏ hàng, xóa sản phẩm khỏi giỏ hàng
- Người dùng có thể xem được giá tạm tính của các sản phẩm trong giỏ hàng
- Người dùng có thể tiến hành đặt hàng và thanh toán từ giỏ hàng

- **Thêm đơn hàng mới:**

- Đơn hàng sẽ được gán một ID duy nhất để theo dõi và quản lý.
- Người dùng có thể thay đổi thông tin nhận hàng như : Tên người nhận, SĐT người nhận, địa chỉ nhận hàng

- **Đặt hàng:**

- Người dùng có thể mua các món ăn đã thêm vào đơn hàng và hoàn tất thanh toán.
- Yêu cầu cung cấp địa chỉ giao hàng và xác nhận thông tin thanh toán.
- Đơn hàng được đánh dấu trạng thái sau khi đặt hàng thành công (ví dụ: đang xử lý, đã giao hàng).

- **Theo dõi đơn hàng:**

- Mỗi người dùng có thể có nhiều đơn hàng khác nhau
- Người dùng có thể theo dõi trạng thái các đơn hàng của mình

- **Đặt chỗ:**

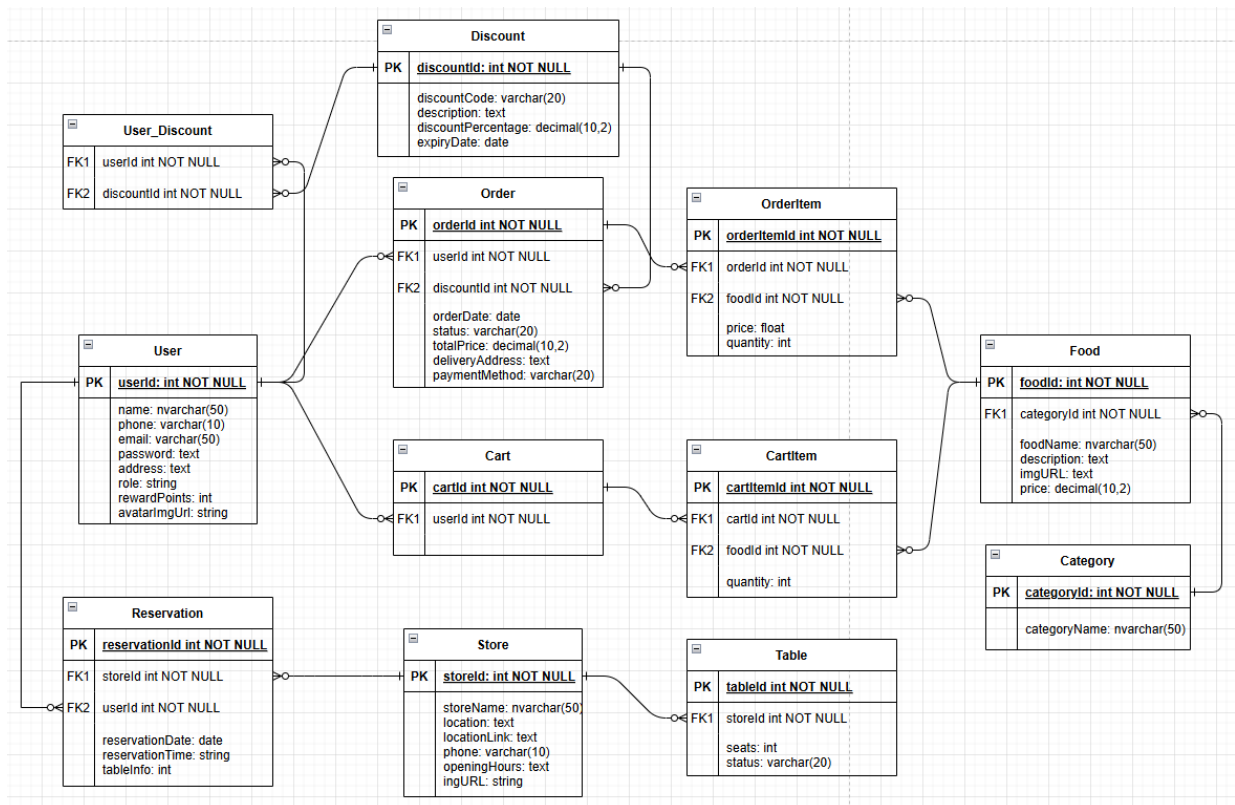
- Người dùng có thể đặt bàn trước tại cửa hàng và điền thông tin như ngày, giờ, ghi chú, chọn bàn mong muốn.
- Mỗi lần đặt chỗ sẽ có ID duy nhất để phân biệt và theo dõi.
- Người dùng có thể xem lại danh sách các đặt chỗ đã thực hiện, truy xuất vị trí cơ sở hay gọi điện cho cửa hàng.

- **Lưu trữ dữ liệu:**

- Ứng dụng cần lưu trữ dữ liệu về người dùng, đơn hàng, đặt chỗ và món ăn vào cơ sở dữ liệu (Firestore) để duy trì dữ liệu ngay cả khi tắt và mở lại ứng dụng.
- Dữ liệu cần được cập nhật theo thời gian thực (Firestore Realtime Database)
- Ứng dụng cần lưu trữ được ảnh của người dùng, các món ăn,... (Firestore Storage)
- Các bản ghi sẽ bao gồm các thông tin liên quan của từng chức năng, giúp dễ dàng truy xuất và cập nhật.

2.2. Thiết kế hệ thống:

2.2.1 Biểu đồ lớp cho mô hình miền (ER Diagram):



Hình 1 : Biểu đồ lớp cho mô hình miền (ER Diagram)

2.2.2 Biểu đồ lớp (Class Diagram):

Thiết kế kiến trúc (MVC):

- **Model:**
 - **User**: Class đại diện cho người dùng, chứa các thuộc tính: *Id*, *họ tên*, *SĐT*, *email*, *mật khẩu*, *địa chỉ*, *vai trò*, *điểm thưởng* và *ảnh*.
 - **Store**: Class đại diện cho cơ sở cửa hàng, chứa các thuộc tính: *Id*, *tên cơ sở*, *địa chỉ*, *link địa chỉ*, *SĐT quán*, *giờ mở cửa* và *link ảnh cơ sở*.
 - **Food**: Class đại diện cho các sản phẩm, chứa các thuộc tính: *Id*, *danh mục*, *tên sản phẩm*, *mô tả*, *link ảnh sản phẩm*, *giá tiền*, *số lượng đã bán*.
 - **Category**: Class đại diện cho danh mục của sản phẩm, chứa các thuộc tính: *Id*, *tên danh mục*.
 - **Cart**: Class đại diện cho giỏ hàng của người dùng, chứa các thuộc tính: *Id*, *Id người dùng*.
 - **CartItem**: Class đại diện cho các sản phẩm có trong giỏ hàng của người dùng, chứa các thuộc tính: *Id*, *Id giỏ hàng*, *Id của sản phẩm*, *số lượng của sản phẩm trong giỏ hàng*.

- **Order:** Class đại diện cho đơn đặt hàng của người dùng, chứa các thuộc tính: *Id, Id người dùng, tên người nhận, SĐT người nhận, địa chỉ nhận hàng, ngày đặt hàng, tổng tiền của đơn hàng, phương thức thanh toán, mã giảm giá được sử dụng*
- **OrderItem:** Class đại diện cho các sản phẩm đã được đặt trong đơn hàng, chứa các thuộc tính: *Id, Id đơn hàng, Id sản phẩm đã đặt, giá của sản phẩm đã đặt, số lượng sản phẩm đã đặt*
- **Reservation:** Class đại diện cho bàn đã được đặt, chứa các thuộc tính: *Id, Id người dùng, Id cơ sở, Id của bàn được đặt, ngày đặt bàn, thời gian đặt bàn, ghi chú.*
- **Store:** Class đại diện cho cơ sở của hàng, chứa các thuộc tính: *Id, tên cơ sở, địa chỉ, link địa chỉ, SĐT quán, giờ mở cửa và link ảnh cơ sở, danh sách bàn của quán.*
- **TableInfo:** Class đại diện cho thông tin của bàn trong quán, chứa các thuộc tính: *Id, Id cửa hàng, số chỗ ngồi, trạng thái bàn.*
- **Discount:** Class đại diện cho mã giảm giá, chứa các thuộc tính: *Id, mã giảm, mô tả, hệ số giảm và ngày hết hạn.*
- **User_Discount:** Class đại diện cho quan hệ giữa người dùng và mã giảm giá, chứa các thuộc tính: *Id người dùng, Id mã giảm giá, ngày nhận và ngày hết hạn.*
- **View (layout):**
 - **activity_home:** Giao diện chính của ứng dụng, gồm có thanh toolbar, thanh điều hướng dưới để chuyển đổi giữa các giao diện chức năng.
 - **fragment_home:** Giao diện Trang chủ, hiển thị danh sách các sản phẩm.
 - **fragment_food_detail:** Giao diện hiển thị những thông tin chi tiết của sản phẩm, gồm ảnh, tên, mô tả, giá tiền và nút “Mua”
 - **fragment_category:** Giao diện hiển thị danh sách sản phẩm lọc theo danh mục, gồm có thanh tìm kiếm, các nút chọn danh mục và danh
 - **fragment_book_table:** Giao diện cho chức năng đặt bàn, gồm 2 tab đặt bàn và xem bàn đã đặt.
 - **fragment_store:** Giao diện hiển thị danh sách các cơ sở của cửa hàng, mỗi cơ sở sẽ có ảnh, tên, giờ mở cửa, số bàn còn trống và nút “Đặt bàn”.
 - **fragment_table_booking:** Giao diện cho việc đặt bàn, gồm tên cơ sở, ngày đặt, giờ đặt, bàn được chọn, ghi chú và nút “Đặt bàn”
 - **fragment_reservation:** Giao diện hiển thị danh sách những bàn đã đặt của người dùng, gồm ảnh và tên cơ sở, ngày giờ đặt bàn, thông tin bàn đặt.

- **fragment_reservaton_detail**: Giao diện hiển thị chi tiết thông tin bàn đã đặt, gồm ảnh và tên cơ sở, ngày giờ đặt, thông tin bàn, ghi chú, các icon để cho chức năng lấy địa chỉ cửa hàng hay gọi cửa hàng và nút “Huỷ đặt bàn”.
- **fragment_cart**: Giao diện hiển thị giỏ hàng của người dùng, gồm có những sản phẩm người dùng đã đặt, thông tin tổng tiền, mã giảm giá và nút “Thanh toán”.
- **fragment_discount**: Giao diện hiển thị danh sách mã giảm giá của người dùng, gồm các mã giảm giá và nút “Chọn”.
- **fragment_profile**: Giao diện hồ sơ của người dùng, gồm có họ tên, số điện thoại, địa chỉ.
- **fragment_edit_profile**: Giao diện chỉnh sửa thông tin người dùng, gồm các ô họ tên, SĐT, địa chỉ, mật khẩu hiện tại, mật khẩu mới, nhập lại mật khẩu mới và nút “Xác nhận”.
- **activity_login**: Giao diện Đăng nhập, gồm logo ứng dụng, ô điền SĐT, mật khẩu, nút “Quên mật khẩu”, “Đăng nhập” và “Đăng ký”.
- **activity_register**: Giao diện chỉnh sửa thông tin người dùng, gồm các ô họ tên, SĐT, địa chỉ, mật khẩu, xác nhận mật khẩu và nút “Đăng ký”.
- **Controller (Fragment và Activity)**
 - **MainActivity**: Xác thực và điều hướng người dùng dựa trên thông tin đăng nhập.
 - **HomeAvtivity**: Cung cấp điều hướng linh hoạt và quản lý các fragment hiển thị
 - **RegisterActivity**: Đăng ký người dùng mới và lưu dữ liệu trên Firebase
 - **BookTableFragment**: Xử lý chức năng chuyển đổi giữa 2 tab đặt bàn và xem bàn đã đặt.
 - **CartFragment**: Xử lý các chức năng của giỏ hàng
 - **CategoryFragment**: Quản lý và hiển thị danh mục cùng món ăn, cho phép người dùng tìm kiếm món ăn theo từ khóa và tải dữ liệu từ Firebase theo danh mục đã chọn.
 - **DetailFoodFragment**: Quản lý hiển thị thông tin chi tiết của món ăn, cho phép người dùng tăng giảm số lượng và thêm món ăn vào giỏ hàng
 - **DetailReservationFragment**: Quản lý hiển thị thông tin chi tiết của đặt bàn, cho phép người dùng hủy đặt bàn.
 - **DiscountFragment**: Cung cấp chức năng quản lý và hiển thị các mã giảm giá cho người dùng, cho phép họ dễ dàng chọn và xác nhận mã giảm giá mà họ muốn sử dụng

- **HomeFragment:** Cho phép người dùng dễ dàng xem các danh sách món ăn như Bán chạy, Món mới, Combo và tìm kiếm các món ăn
- **OrderFragment:** Cho phép người dùng xác nhận và đặt hàng, đồng thời quản lý thông tin giao hàng như : Họ tên người nhận, SĐT người nhận, địa chỉ nhận hàng
- **OrderViewFragment:** Cho phép người dùng xem danh sách các đơn hàng của họ
- **ProfileFragment:** Cho phép người dùng xem và chỉnh sửa thông tin cá nhân của họ. Nó cũng hỗ trợ chức năng đăng xuất với xác nhận
- **ReservationFragment:** Hiện thị danh sách đặt chỗ của người dùng
- **SelectLocationFragment:** Sử dụng Mapbox để xây dựng một màn hình cho phép người dùng chọn vị trí trên bản đồ
- **StoreFragment:** Giúp người dùng dễ dàng tìm kiếm các cửa hàng và đặt bàn một cách nhanh chóng và thuận tiện, đồng thời cung cấp thông tin rõ ràng về tình trạng bàn trống của từng cửa hàng
- **TableBookingFragment:** Giúp người dùng dễ dàng chọn ngày, giờ và bàn để đặt tại các cửa hàng, đồng thời quản lý các yêu cầu đặt bàn

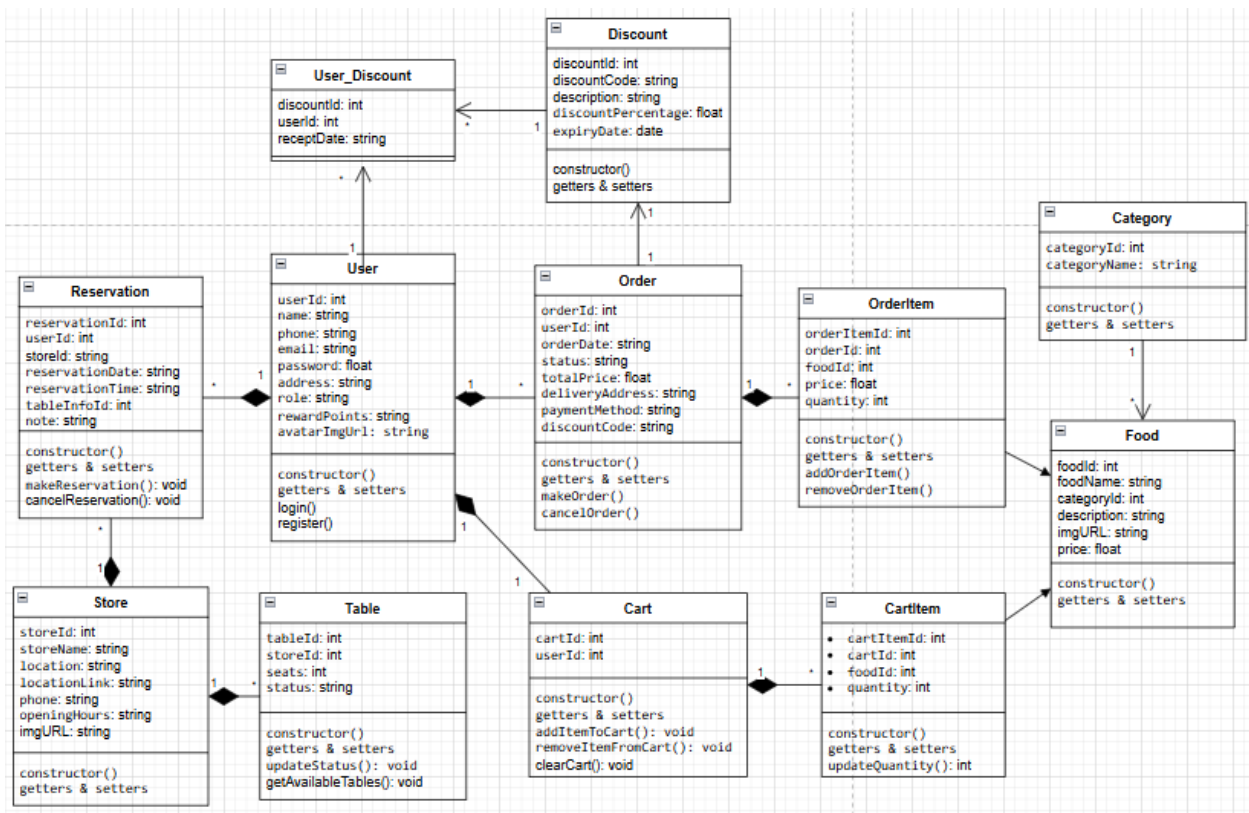
Dựa vào kiến trúc trên, ta có thể xác định các lớp sau:

- **Lớp User:**
 - **Thuộc tính:** userId, name, phone ,email, password, address, role, rewardPoints.
 - **Phương thức:**
 - Constructor().
 - Getters và setters cho các thuộc tính.
 - login()
 - register()
- **Lớp Food**
 - **Thuộc tính:** foodId, foodName, categoryId, description, imgUrl, price.
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
- **Lớp Category**

- **Thuộc tính:** categoryId, categoryName
- **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
- **Lớp Cart**
 - **Thuộc tính:** cartId, userId
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - addFoodToCart()
 - removeFoodFromCart()
 - clearCart()
- **Lớp CartItem**
 - **Thuộc tính:** cartItemId, cartId, foodId, quantity
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - updateQuantity()
- **Lớp Order**
 - **Thuộc tính:** orderId, userId, orderDate, status, totalPrice, deliveryAddress, paymentMethod, discountCode.
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - makeOrder()
 - cancelOrder()
- **Lớp OrderItem**
 - **Thuộc tính:** orderItemId, orderId, foodId, price, quantity

- **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - addOrderItem()
 - removeOrderItem()
- **Lớp Store**
 - **Thuộc tính:** storeId, storeName, location, locationLink, phone, openingHours, imgUrl.
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
- **Lớp TableInfo**
 - **Thuộc tính:** tableInfoId, storeId, seats, status
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - updateStatus()
 - getAvailableTables()
- **Lớp Reservation**
 - **Thuộc tính:** reservationId, userId, storeId, tableInfoId, reservationDate, reservationTime, note.
 - **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
 - makeReservation()
 - cancelReservation()
- **Lớp Discount**
 - **Thuộc tính:** discountId, discountCode, description, discountPercentage, expiryDate

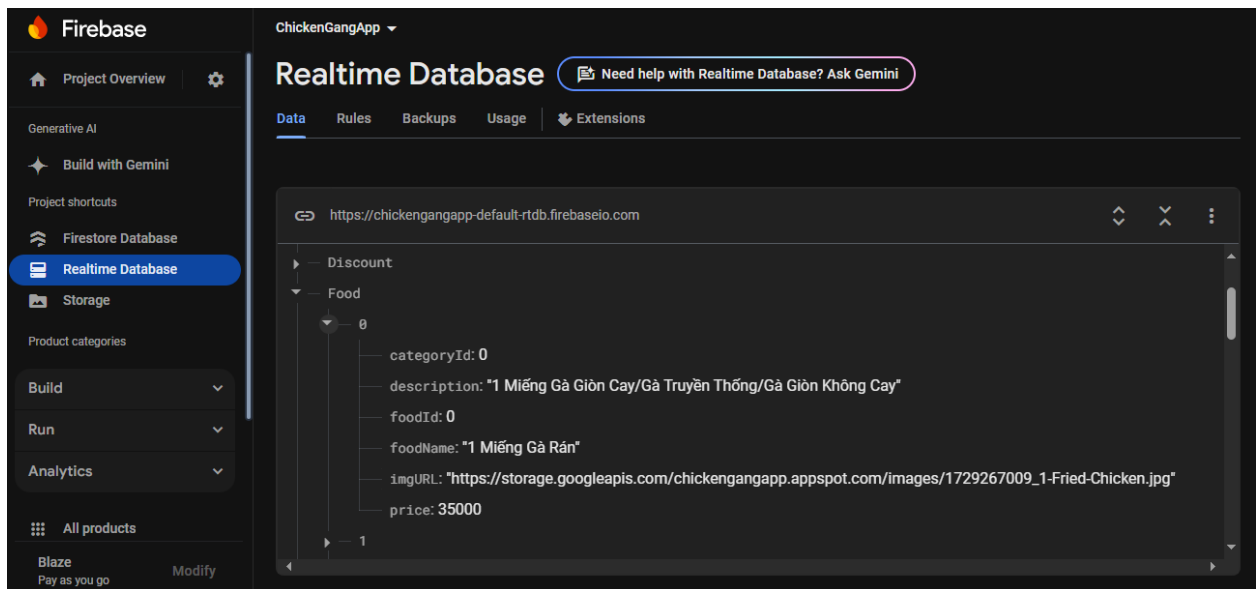
- **Phương thức:**
 - Constructor()
 - Getters và setters cho các thuộc tính.
- **Lớp User_Discount**
 - **Thuộc tính:** discountId, userId, receiptDate.
 - **Phương thức** (không có)



Hình 2: Biểu đồ lớp (Class Diagram)


2.3 Thiết kế cơ sở dữ liệu:

- Tạo một Firebase Realtime Database để lưu trữ dữ liệu.
- Tạo Firebase Storage để lưu trữ ảnh
- Xác định tên key và các value tương ứng với các thuộc tính của công việc.



Hình 3: Cấu trúc database

2.4 Thiết kế giao diện:




Số điện thoại

Mật khẩu

[Quên mật khẩu ?](#)

Đăng nhập

Bạn chưa có tài khoản ? [Đăng ký ngay!](#)



Họ tên

Số điện thoại

Địa chỉ

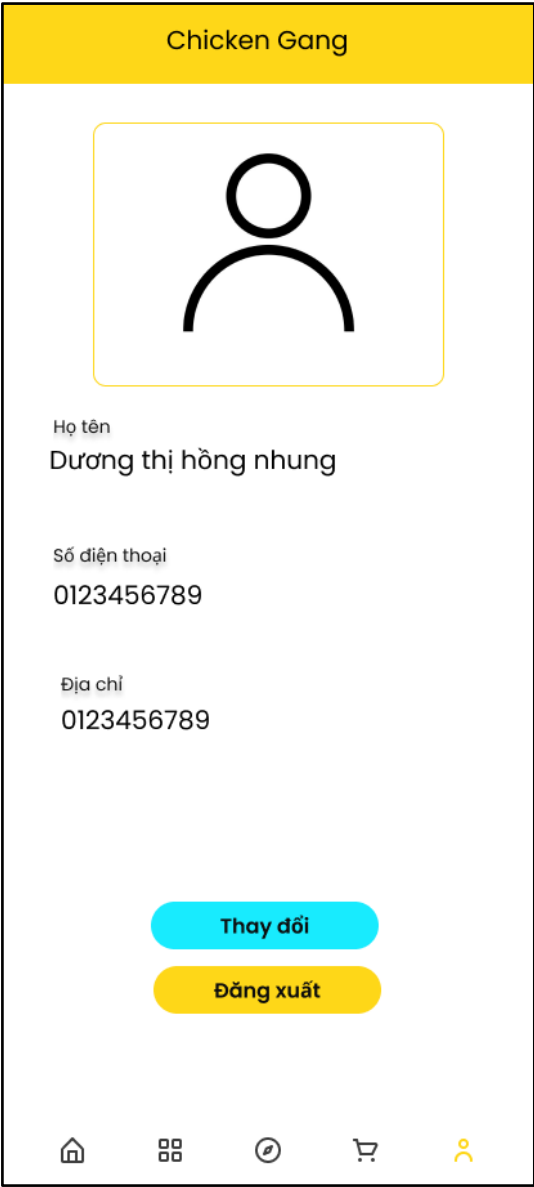
Mật khẩu

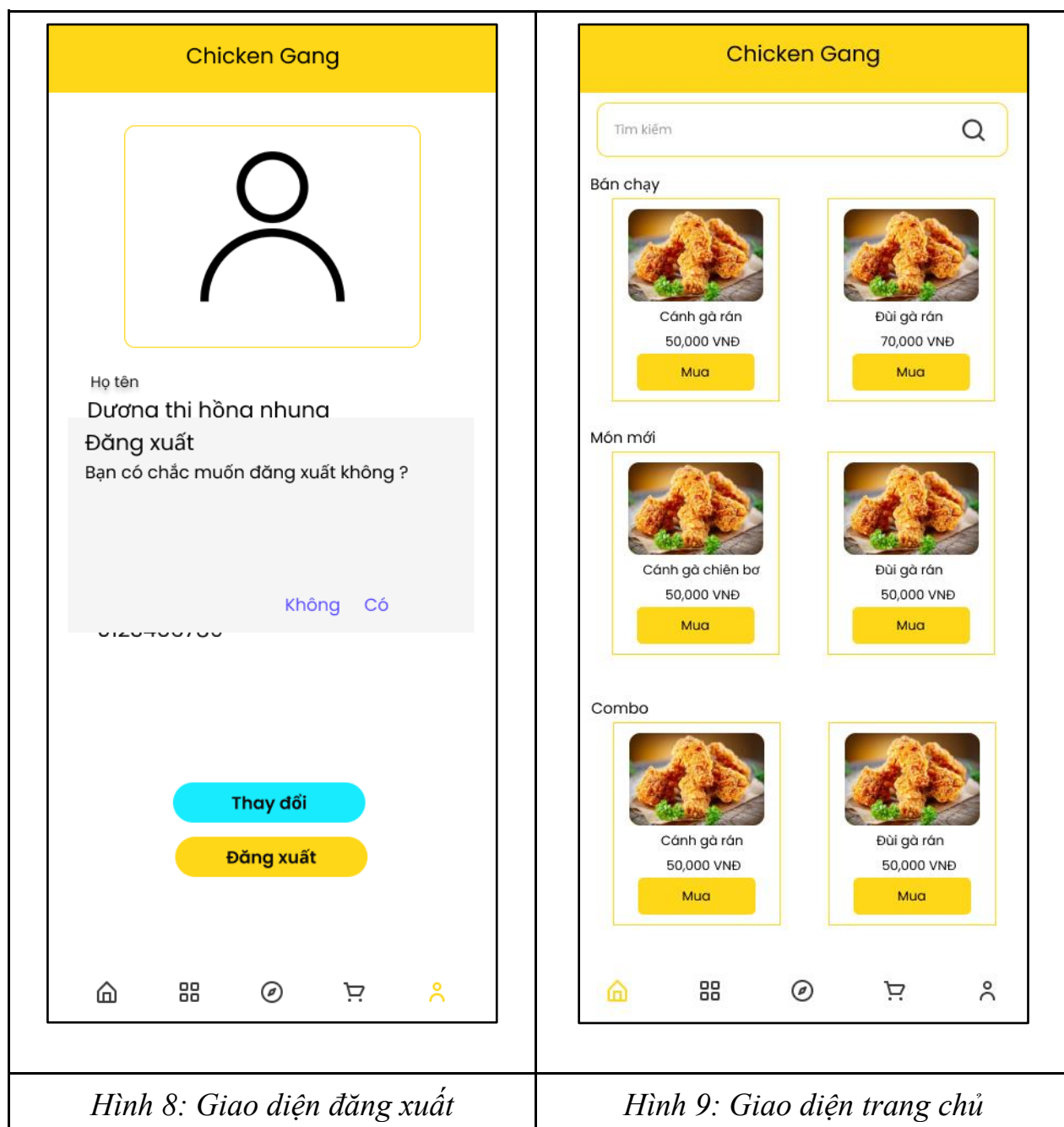
Xác nhận mật khẩu

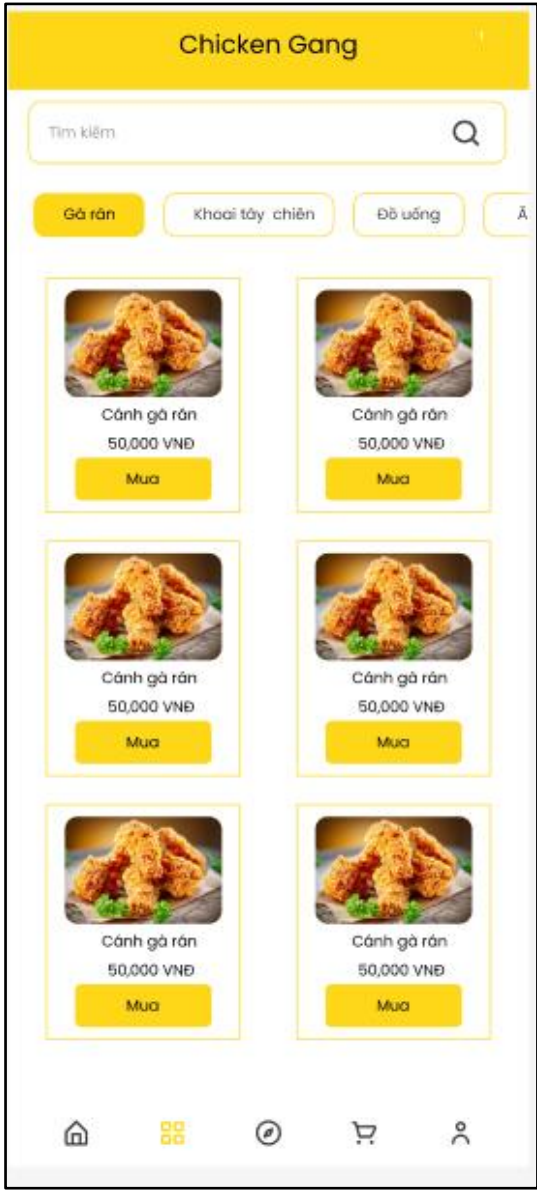
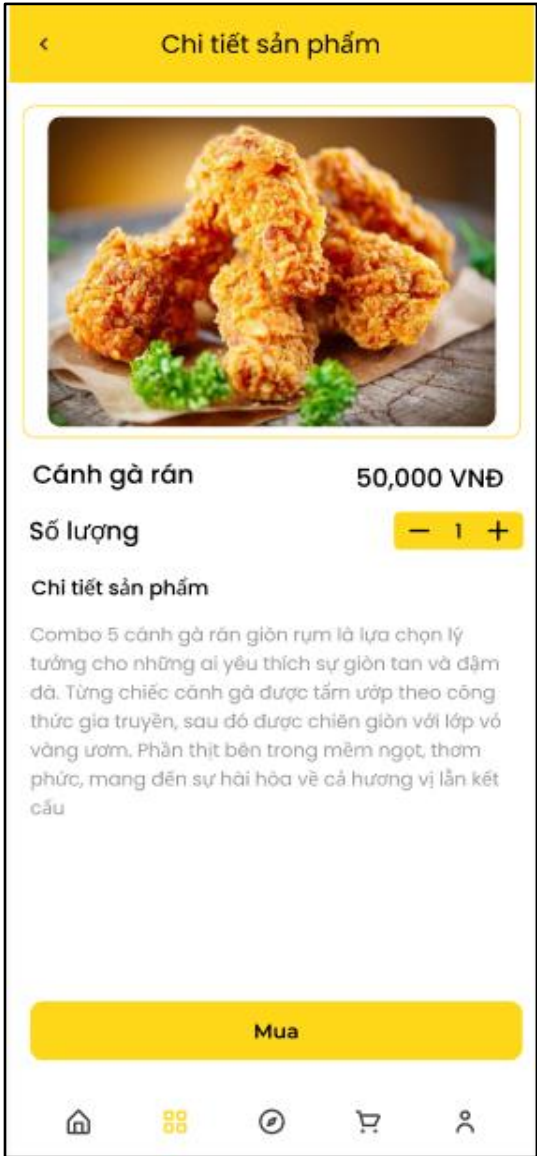
Đăng ký

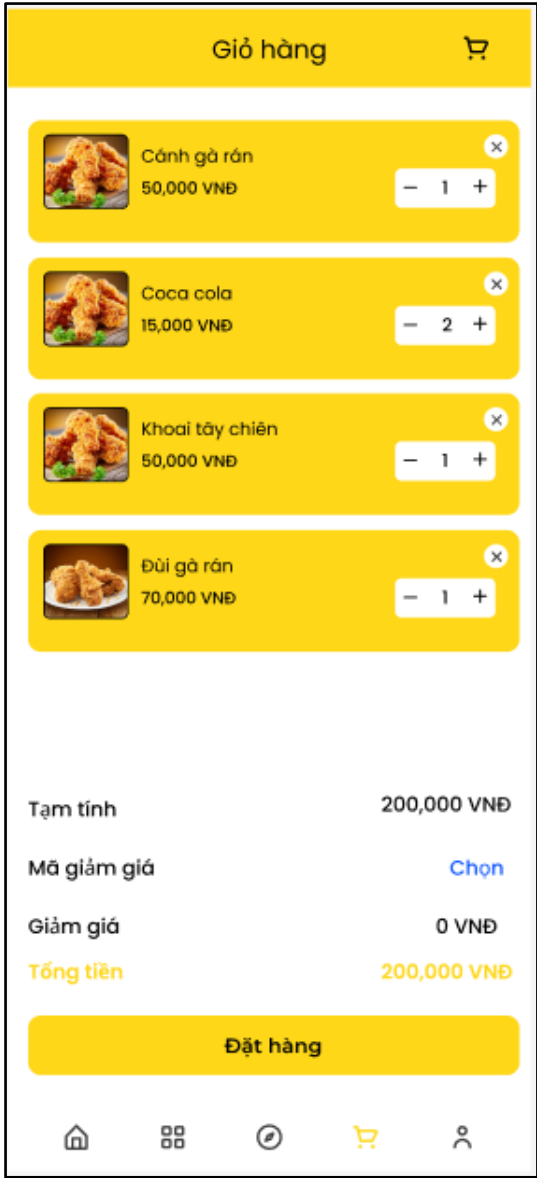

Bạn đã có tài khoản ? [Đăng nhập](#)


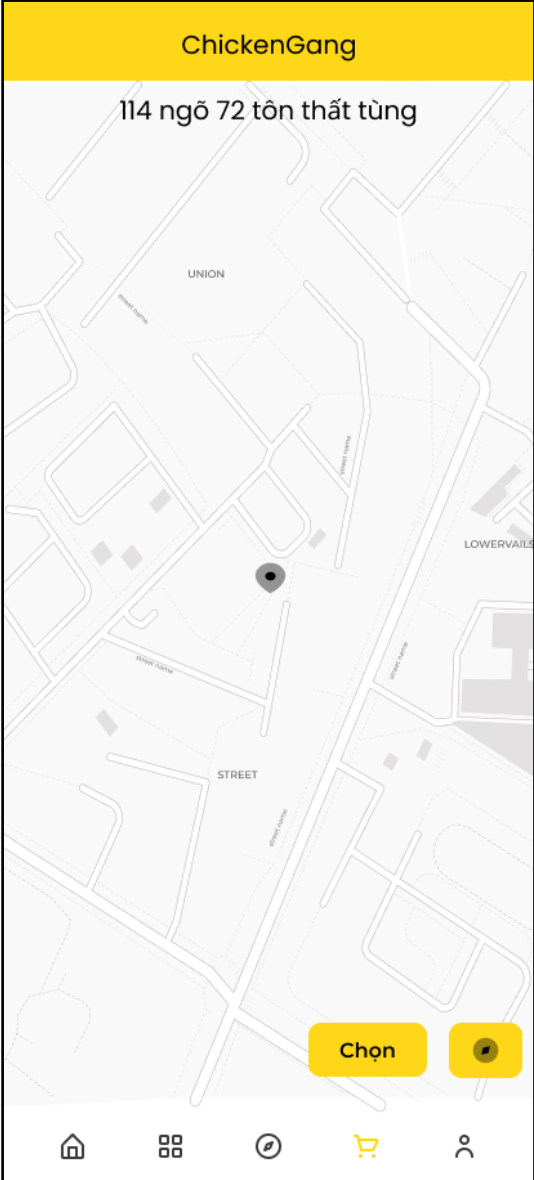
<i>Hình 4: Giao diện đăng nhập</i>	<i>Hình 5: Giao diện đăng ký</i>

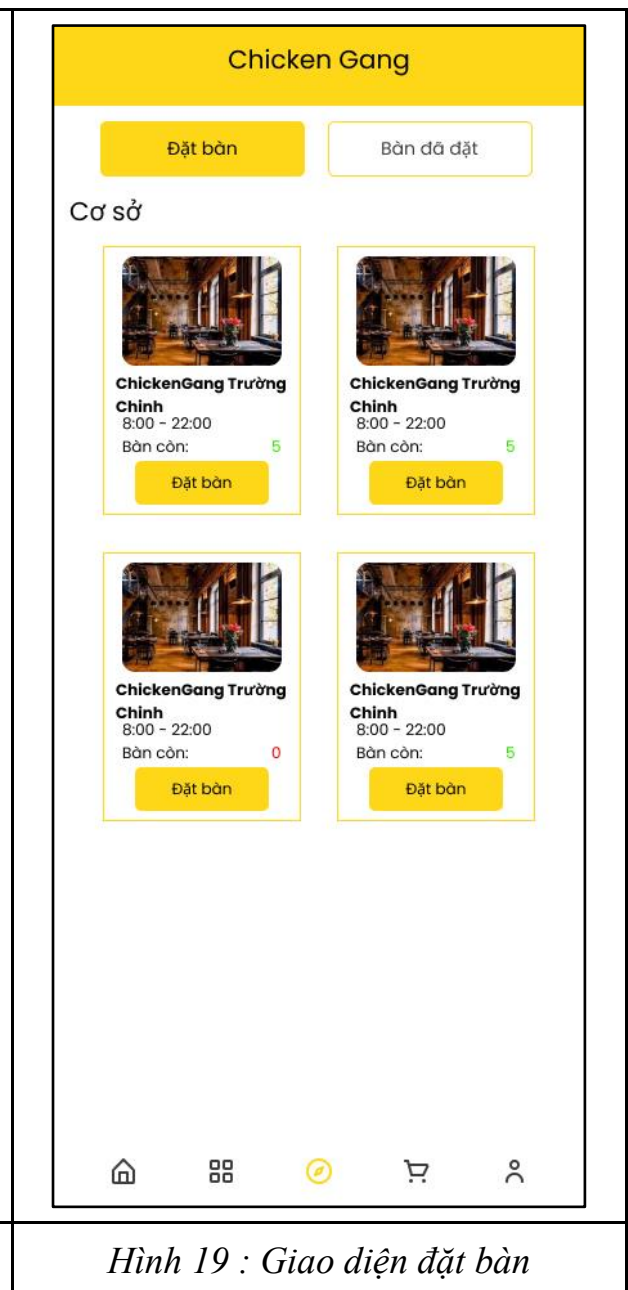
	
<p>Hình 6: Giao diện xem thông tin cá nhân</p>	<p>Hình 7: Giao diện chỉnh sửa thông tin cá nhân</p>








	
<p>Hình 10: Giao diện danh mục</p>	<p>Hình 11: Giao diện chi tiết sản phẩm</p>

 <p>Giỏ hàng</p> <p>Cánh gà rán 50,000 VNĐ</p> <p>Coca cola 15,000 VNĐ</p> <p>Khoai tây chiên 50,000 VNĐ</p> <p>Đùi gà rán 70,000 VNĐ</p> <p>Tạm tính 200,000 VNĐ</p> <p>Mã giảm giá Chọn</p> <p>Giảm giá 0 VNĐ</p> <p>Tổng tiền 200,000 VNĐ</p> <p>Đặt hàng</p>	 <p>Giảm Giá</p> <p>NEWMEM10 Giảm giá 10 % cho khách hàng mới</p> <p>NEWMEM20 Giảm giá 20 % cho khách hàng mới</p> <p>NEWMEM15 Giảm giá 15 % cho khách hàng mới</p> <p>NEWMEM50 Giảm giá 50 % cho khách hàng mới</p>
<p>Hình 12: Giao diện giỏ hàng</p>	<p>Hình 13: Giao diện chọn mã giảm giá</p>

	
<p>Hình 14: Giao diện xác nhận thông tin thanh toán</p>	<p>Hình 15: Giao diện chọn vị trí nhận hàng</p>



<div data-bbox="248 165 786 1346"> <div> <div><</div> <div>Đặt bàn</div> </div> <div> <div> <div>ChickentGang Trường Chinh</div> <div>Đường đi</div> </div> <div>Chinh</div> </div> <div> <div>Ngày đặt bàn</div> <div>Chọn ngày</div> </div> <div> <div>Giờ đặt bàn</div> <div>Chọn giờ</div> </div> <div> <div>Chọn bàn</div> <div> <div>4 chỗ</div> <div>2 chỗ</div> <div>6 chỗ</div> <div>2 chỗ</div> </div> </div> <div> <div>Ghi chú</div> <div></div> </div> <div>Xác nhận</div> <div> <div>🏠</div> <div>🍽️</div> <div>🕒</div> <div>🛒</div> <div>👤</div> </div> </div>	<div data-bbox="866 165 1404 1346"> <div>Chicken Gang</div> <div> <div>Đặt bàn</div> <div>Bàn đã đặt</div> </div> <div>Bàn đã đặt</div> <div> <div> <div>  <div> <div>ChickentGang Trường Chinh</div> <div>Ngày : 07/10/2024</div> <div>Giờ : 10:00</div> <div>Bàn: 4 chỗ</div> </div> </div> <div> <div>  <div> <div>ChickentGang Trường Chinh</div> <div>Ngày : 07/10/2024</div> <div>Giờ : 10:00</div> <div>Bàn: 4 chỗ</div> </div> </div> <div> <div>  <div> <div>ChickentGang Trường Chinh</div> <div>Ngày : 07/10/2024</div> <div>Giờ : 10:00</div> <div>Bàn: 4 chỗ</div> </div> </div> <div> <div>  <div> <div>ChickentGang Trường Chinh</div> <div>Ngày : 07/10/2024</div> <div>Giờ : 10:00</div> <div>Bàn: 4 chỗ</div> </div> </div> </div> <div> <div>🏠</div> <div>🍽️</div> <div>🕒</div> <div>🛒</div> <div>👤</div> </div> </div> </div></div></div></div>
<div>Hình 20: Giao diện chi tiết đặt bàn</div>	<div>Hình 21: Giao diện bàn đã đặt</div>

<div data-bbox="248 165 786 1344"> <div data-bbox="280 192 592 221"> < Bàn đã đặt </div> <div data-bbox="280 257 753 519">  </div> <div data-bbox="272 555 764 624"> <div> ChickentGang Trường Chinh Đường đi </div> </div> <div data-bbox="272 651 606 685"> <p>Số điện thoại :0123456789</p> </div> <div data-bbox="272 725 510 763"> <p>Ngày : 07/10/2024</p> </div> <div data-bbox="272 804 405 837"> <p>Giờ : 10:00</p> </div> <div data-bbox="272 878 414 911"> <p>Bàn: 4 chỗ</p> </div> <div data-bbox="272 952 743 1025"> <p>Ghi chú Ghi chúGhi chúGhi chúGhi chú chúGhi chú</p> </div> <div data-bbox="272 1133 730 1167"> <p>Lưu ý : Bàn chỉ được giữ trong vòng 15 phút!</p> </div> <div data-bbox="272 1189 774 1249"> <div>Hủy đặt bàn</div> </div> <div data-bbox="304 1288 726 1321"> <div> 🏠 🍽️ 🕒 🛒 👤 </div> </div> </div>	
<p>Hình 22: Giao diện chi tiết bàn đã đặt</p>	

2.3. Triển khai:

- **Viết code:** Sử dụng ngôn ngữ Java để xây dựng các lớp của dự án, đọc/ghi dữ liệu bằng Firebase Realtime Database, xử lý dữ liệu và hiển thị lên giao diện ứng dụng.
- **Kiểm thử:** Viết các test case để kiểm tra các chức năng của ứng dụng, đảm bảo các chức năng của ứng dụng hoạt động đúng theo yêu cầu.

2.4. Vận hành và bảo trì:

- **Cài đặt và triển khai:**
 - Hướng dẫn chi tiết người dùng về cách cài đặt ứng dụng, bao gồm yêu cầu hệ thống, các bước cài đặt từ cửa hàng ứng dụng hoặc thông qua file APK.
 - Đảm bảo triển khai ứng dụng lên các nền tảng phân phối như Google Play Store để người dùng có thể tải và cập nhật dễ dàng.
- **Bảo trì:**
 - **Sửa lỗi phát sinh:** Thường xuyên kiểm tra và khắc phục các lỗi người dùng báo cáo để đảm bảo trải nghiệm mượt mà và ổn định.
 - **Cập nhật chức năng mới:** Dựa trên phản hồi từ người dùng và nhu cầu thực tế, bổ sung hoặc nâng cấp các tính năng để đáp ứng tốt hơn yêu cầu của người dùng.
 - **Cải thiện hiệu năng:** Tối ưu hóa ứng dụng về tốc độ, dung lượng và tiêu thụ tài nguyên nhằm mang đến trải nghiệm tốt hơn cho người dùng trên nhiều loại thiết bị.

CHƯƠNG 3. KẾT QUẢ THỰC HIỆN

3.1. Công nghệ đã sử dụng

- Ngôn ngữ lập trình: Java
- Công cụ: Android Studio
- Thư viện: có thể sử dụng các thư viện hỗ trợ đọc/ghi file, xử lý dữ liệu (các thư viện của Firebase, Glide, ...)

3.2. Tiến độ thực hiện

Link github tới dự án: [DuyTaiNguyxn/CSE441_PROJECT: Nguyễn Duy Tài, Lê Văn Bình, Dương Thị Hồng Nhung \(github.com\)](https://github.com/DuyTaiNguyxn/CSE441_PROJECT)

Hướng dẫn các bước đã thực hiện:

B1. Tạo dự án mới:

- Mở Android Studio.
- Chọn "Create New Project".
- Nhập tên dự án ("CSE441_PROJECT").
- Thiết lập Package name (ví dụ: "com.duytai.cse441_project")
- Chọn "Java" làm ngôn ngữ lập trình.
- Chọn Minimum SDK phù hợp (ví dụ: API 24 ("Nougat"; Android 7.0) hoặc mới hơn).
- Chọn vị trí lưu trữ dự án.
- Nhấn "Finish" để tạo dự án.

B2. Tạo các package:

- Trong cửa sổ "Android", click chuột phải vào thư mục "app/com/duytai/cse441_project".
- Chọn "New" -> "Package".
- Tạo các package sau: model, fragment, adapter

B3. Tạo các lớp:

Cùng với MainActivity, tạo thêm các lớp HomeActivity, RegisterActivity tương ứng cho giao diện Đăng nhập, Trang chủ và Đăng ký.

Trong mỗi package, click chuột phải và chọn "New" -> "Java Class" để tạo các lớp cùng các thuộc tính và phương thức tương ứng:

- **model:** User, Food, Category, Cart, CartItem, Order, OrderItem, Store, TableInfo, Reservation, Discount
- **fragment:** HomeFragment, DetailFoodFragment, CategoryFragment, BookTableFragment, StoreFragment, TableBookingFragment, ReservationFragment, DetailReservationFragment, CartFragment, DiscountFragment, OrderFragment, OrderViewFragment, SelectLocationFragment, ProfileFragment.
- **adapter:** FoodAdapter, CategoryAdapter, BookTableAdapter, StoreAdapter, TableBookingAdapter, ReservationAdapter, CartAdapter, OrderViewAdapter, DiscountAdapter.

B4. Cài đặt thư viện:

- Mở file build.gradle (Module :app).
- Thêm các thư viện cần thiết:

```
plugins {
    alias(libs.plugins.android.application)
    id("com.google.gms.google-services")
    alias(libs.plugins.google.android.libraries.mapsplatform.secrets.gradle.plugin)
}
```

```
dependencies {
    implementation(libs.appcompat)
    implementation(libs.material)
    implementation(libs.activity)
    implementation(libs.constraintlayout)
    implementation(libs.firebase.database)
    implementation(libs.cardview)
    implementation(libs.play.services.maps)
    testImplementation(libs.junit)
    androidTestImplementation(libs.ext.junit)
    androidTestImplementation(libs.espresso.core)
    implementation(platform("com.google.firebase:firebase-bom:33.4.0"))
    implementation("com.google.firebase:firebase-analytics")
    implementation("com.google.firebase:firebase-auth:21.3.0")
    implementation("com.github.bumptech.glide:glide:4.12.0")
    annotationProcessor("com.github.bumptech.glide:compiler:4.12.0")
    implementation("com.squareup.picasso:picasso:2.8")
    implementation("androidx.swiperefreshlayout:swiperefreshlayout:1.2.0-alpha01")
    implementation("com.mapbox.mapboxsdk:mapbox-android-sdk:9.6.2")
    implementation("com.google.android.gms:play-services-wallet:18.0.0")
}
```

- Chọn File — “Sync Project with Gradle Files”

B5. Viết code:

- Khai báo chức năng và quyền trong AndroidManifest:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-feature
        android:name="android.hardware.telephony"
        android:required="false" />

    <uses-permission android:name="android.permission.CALL_PHONE" />
    <uses-permission android:name="android.permission.INTERNET" />

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <meta-data
        android:name="com.google.android.gms.wallet.api.enabled"
        android:value="true" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:enableOnBackInvokedCallback="true"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/logo_circle"
        android:label="ChickenGang">
```

- Bắt đầu viết code cho từng lớp, thực hiện các chức năng của ứng dụng (ví dụ với chức năng Đăng nhập (MainAcvivity))

```
package com.duytai.cse441_project;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class MainActivity extends AppCompatActivity {
    Button btn_login;
```

```

EditText etPhoneNumber, etPassword;
TextView tvPhoneError, tvPasswordError;
private String phoneNumber, password;
private SharedPreferences sharedPreferences;

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

    boolean isValid = true;

    // Lấy SharedPreferences để lưu userId của người dùng hiện tại
    sharedPreferences = getSharedPreferences("currentUserId",
MODE_PRIVATE);

    // Đặt padding cho các view để tránh phần hệ thống (thanh trạng thái,
    thanh điều hướng)
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
        Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
        return insets;
    });

    // Liên kết các view từ layout với các biến
    btn_login = findViewById(R.id.btn_login);
    etPhoneNumber = findViewById(R.id.et_phone_number);
    etPassword = findViewById(R.id.et_password);
    tvPhoneError = findViewById(R.id.tv_phone_error);
    tvPasswordError = findViewById(R.id.tv_password_error);
    Button btnRegister = findViewById(R.id.btn_register);

    //sự kiện khởi chạy RegisterActivity khi nhấn btn đăng ký
    btnRegister.setOnClickListener(v -> {
        Intent intent = new Intent(MainActivity.this,
RegisterActivity.class);
        startActivity(intent);
    });

    // Thiết lập lắng nghe sự kiện nhấn nút đăng nhập
    btn_login.setOnClickListener(v -> {
        phoneNumber = etPhoneNumber.getText().toString().trim();
        password = etPassword.getText().toString().trim();

        // Kiểm tra nếu thông tin đăng nhập hợp lệ thì thực hiện xác minh
với Firebase
        if (validateLogin()) {
            verifyCredentials(phoneNumber, password);
        }
    });

    // Đặt TextWatcher cho số điện thoại và mật khẩu để kiểm tra tính hợp
    lệ
    if (TextUtils.isEmpty(etPhoneNumber.getText().toString())) {
        tvPhoneError.setText("Vui lòng nhập số điện thoại");
        tvPhoneError.setVisibility(View.VISIBLE);
        isValid = false;
    }

```

```

    } else {
        tvPhoneError.setVisibility(View.GONE);
    }

    etPhoneNumber.addTextChangedListener(new
SimpleTextWatcher(etPhoneNumber, tvPhoneError));
    if (TextUtils.isEmpty(etPassword.getText().toString())) {
        tvPasswordError.setText("Vui lòng nhập mật khẩu");
        tvPasswordError.setVisibility(View.VISIBLE);
        isValid = false;
    } else {
        tvPasswordError.setVisibility(View.GONE);
    }
    etPassword.addTextChangedListener(new SimpleTextWatcher(etPassword,
tvPasswordError));
}

// Phương thức xác minh tài khoản và mật khẩu từ Firebase
private void verifyCredentials(String phoneNumber, String password) {
    DatabaseReference usersRef =
FirebaseDatabase.getInstance().getReference("User");

    // Truy vấn Firebase dựa vào số điện thoại
    usersRef.orderByChild("phone").equalTo(phoneNumber).addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            if (snapshot.exists()) {
                for (DataSnapshot userSnapshot : snapshot.getChildren())
                {
                    String storedPassword =
userSnapshot.child("password").getValue(String.class);
                    if (storedPassword != null &&
storedPassword.equals(password)) {
                        Integer userId =
userSnapshot.child("userId").getValue(Integer.class);

                        // Kiểm tra userID không phải null trước khi sử
dụng
                        if (userId != null) {
                            sharedPreferences.edit().putInt("userId",
userId).apply();

                            // Chuyển sang HomeActivity sau khi đăng nhập
thành công
                            Intent intent_login = new
Intent(MainActivity.this, HomeActivity.class);
                            startActivity(intent_login);
                            finish();
                        } else {
                            Toast.makeText(MainActivity.this, "Không tìm
thấy userID", Toast.LENGTH_SHORT).show();
                        }
                    } else {
                        // Thông báo lỗi nếu mật khẩu không đúng
                        tvPasswordError.setText("Sai mật khẩu");
                        tvPasswordError.setVisibility(View.VISIBLE);
                    }
                }
            } else {
                // Thông báo lỗi nếu số điện thoại không tồn tại

```

```

        tvPhoneError.setText("Số điện thoại không tồn tại");
        tvPhoneError.setVisibility(View.VISIBLE);
    }
}

@Override
public void onCancelled(@NonNull DatabaseError error) {
    // Thông báo nếu có lỗi kết nối đến Firebase
    Toast.makeText(MainActivity.this, "Kết nối tới cơ sở dữ liệu
thất bại!", Toast.LENGTH_SHORT).show();
}
});
}

// Phương thức kiểm tra nếu không có lỗi (tức là tất cả thông tin hợp lệ)
private boolean validateLogin() {
    return tvPhoneError.getVisibility() == View.GONE &&
tvPasswordError.getVisibility() == View.GONE;
}
}

```

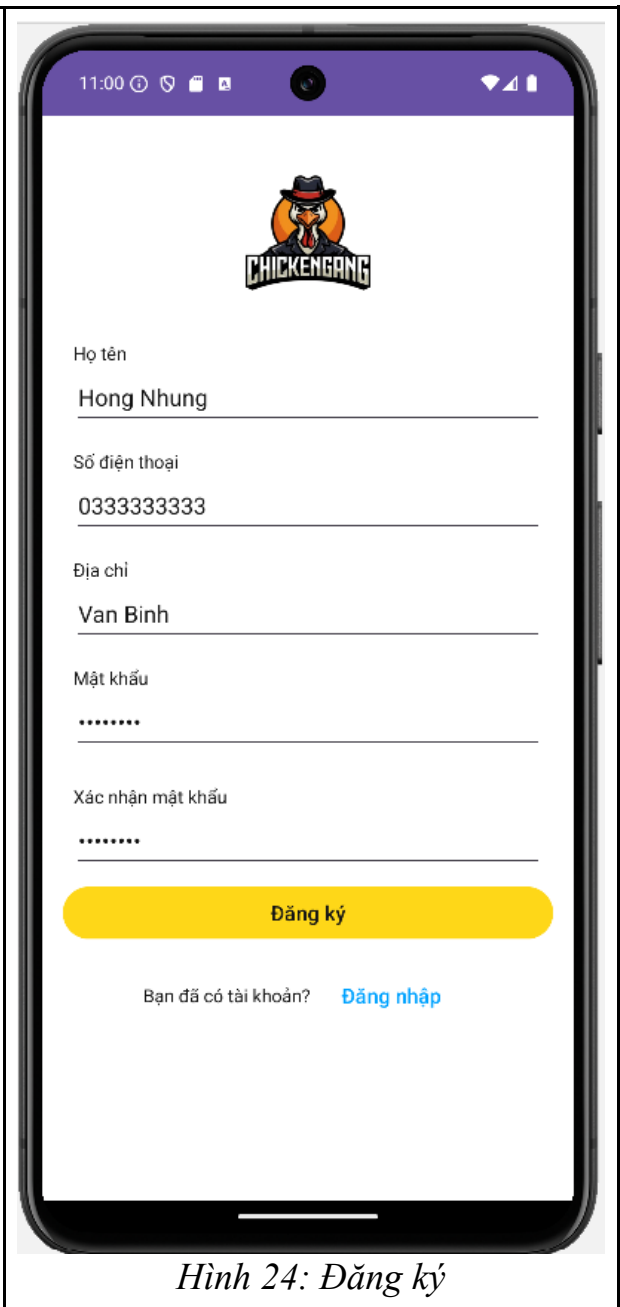
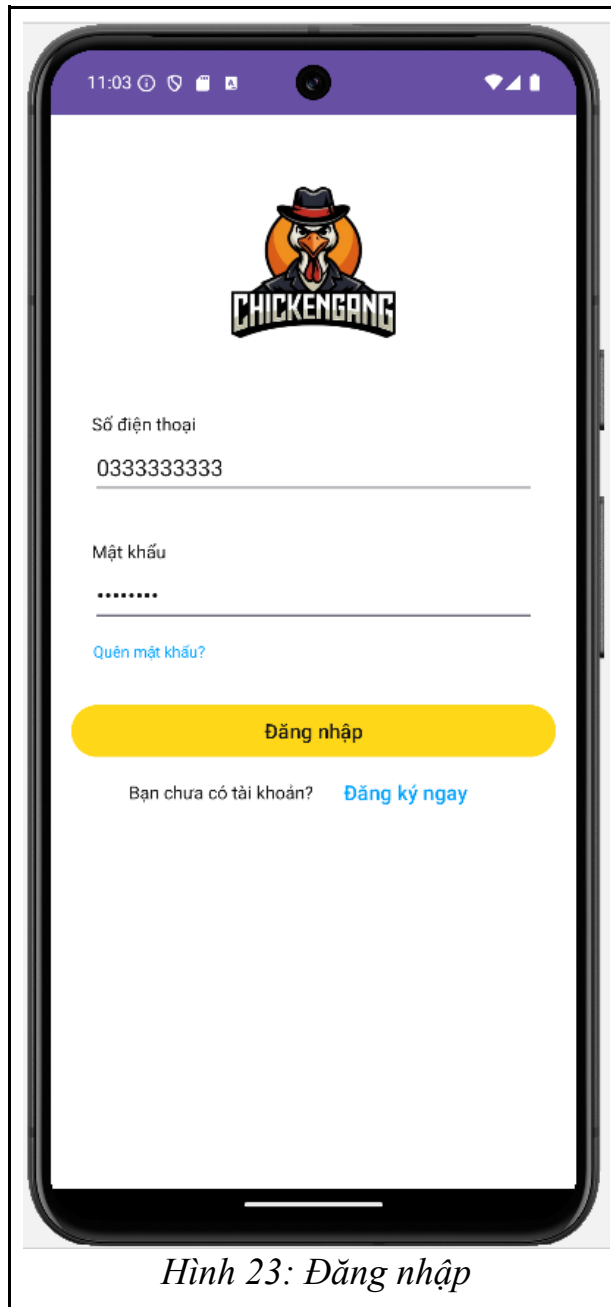
B7. Chạy và kiểm thử:

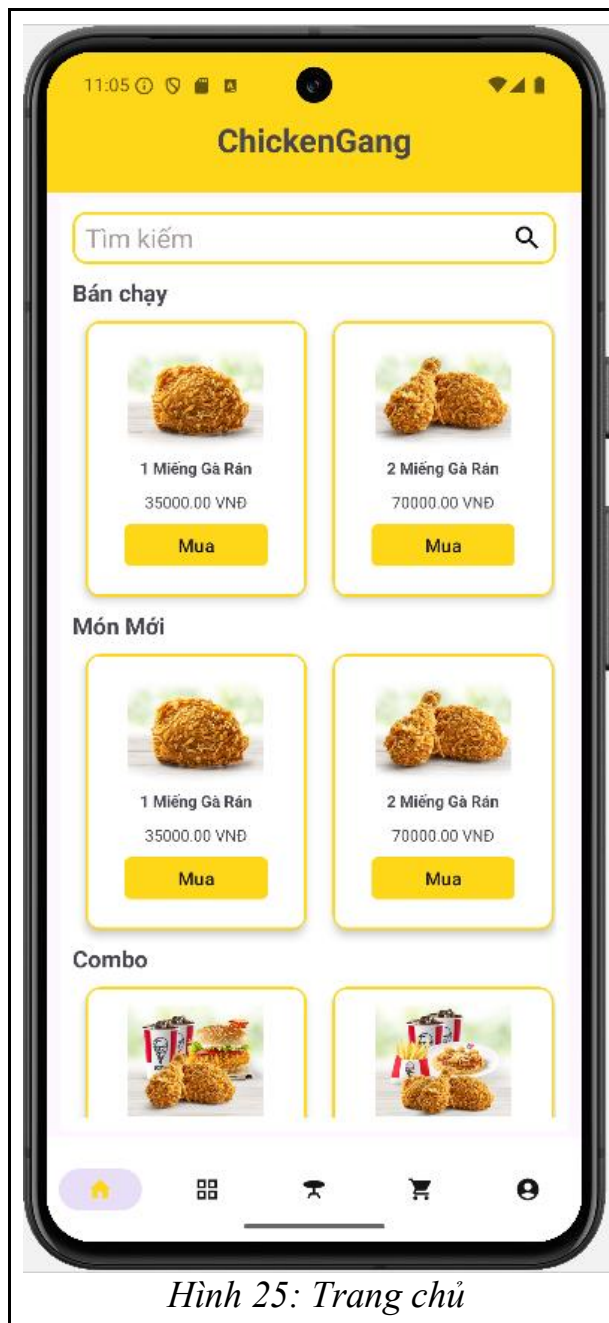
- Chạy thử ứng dụng trên máy ảo của Android S.
- Kiểm tra các chức năng của ứng dụng, sửa lỗi và hoàn thiện code.

B8. Triển khai (tùy chọn):

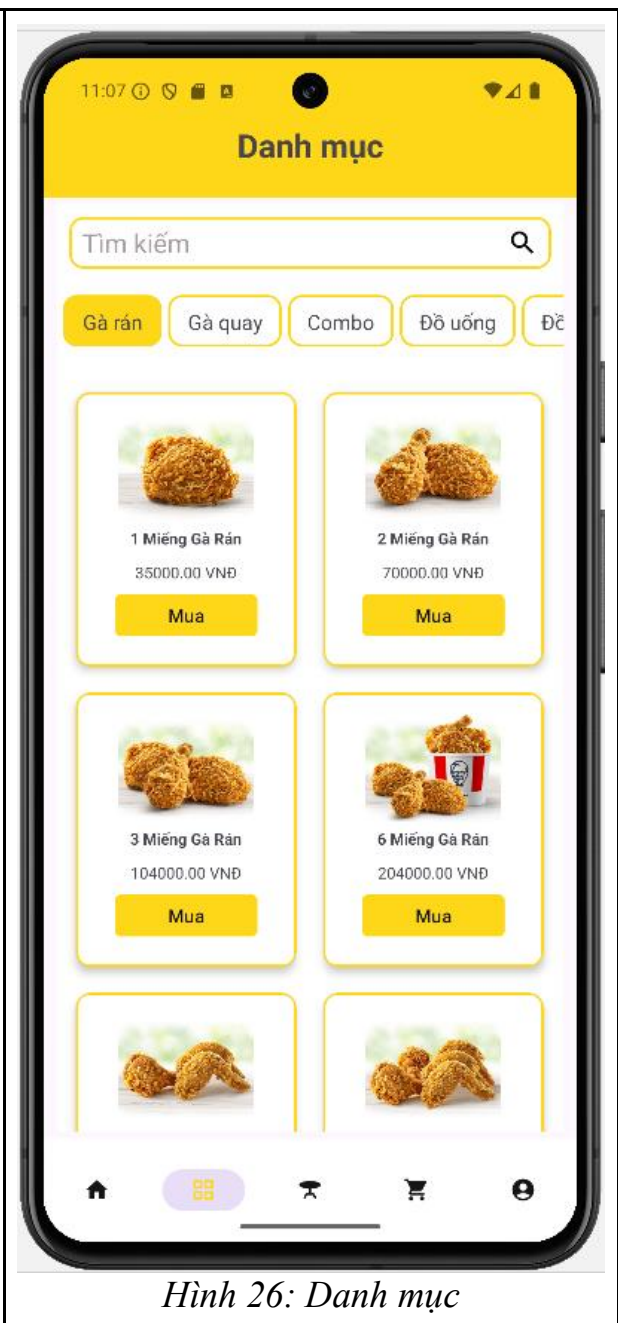
- Có thể đóng gói ứng dụng thành file APK để dễ dàng chia sẻ và chạy trên các máy khác sẽ sử dụng chức năng Build > Build Bundle(s) / APK(s) > Build APK(s) của Android Studio.

3.3. Hình ảnh sản phẩm

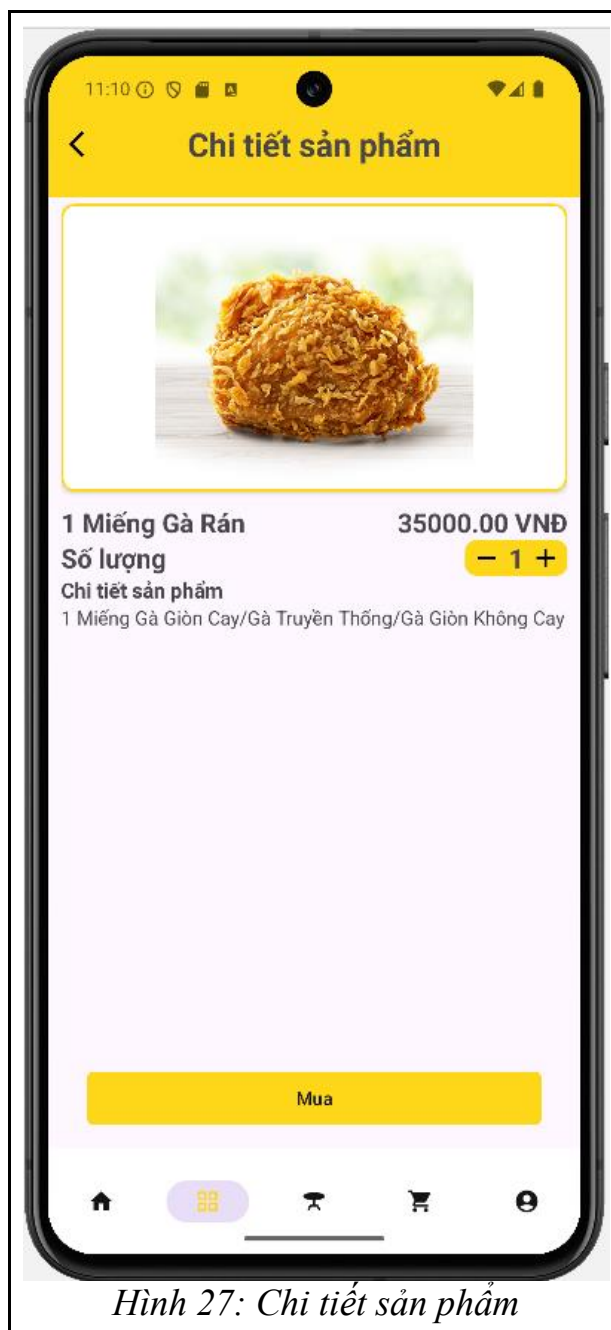




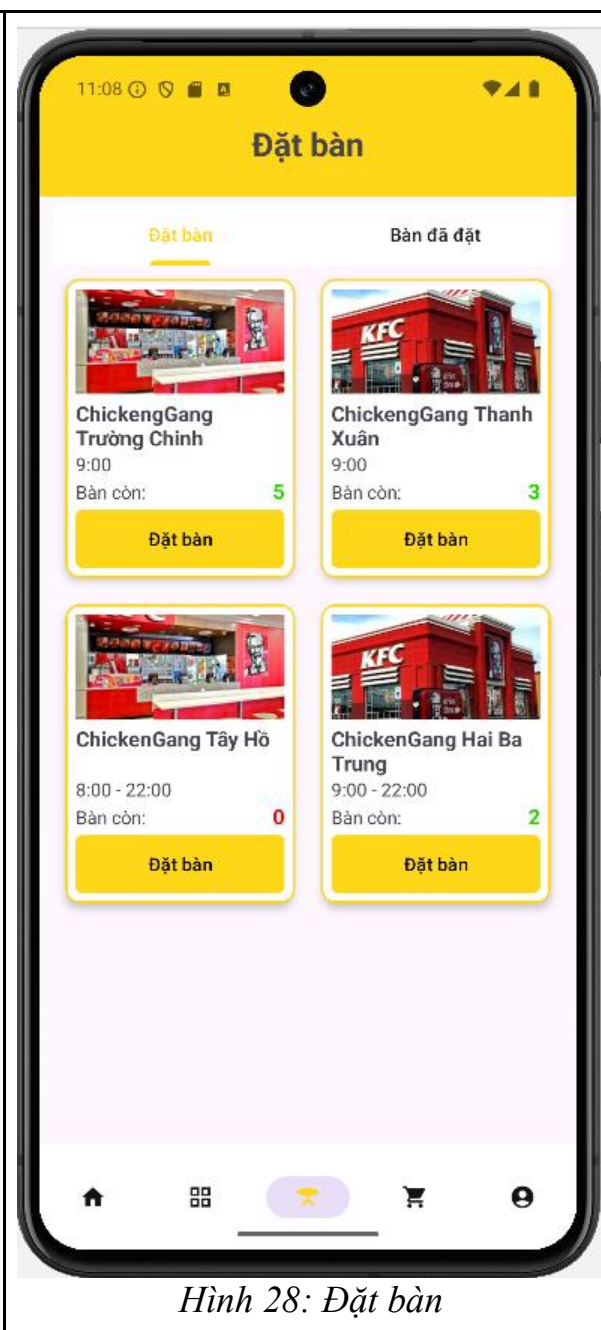
Hình 25: Trang chủ



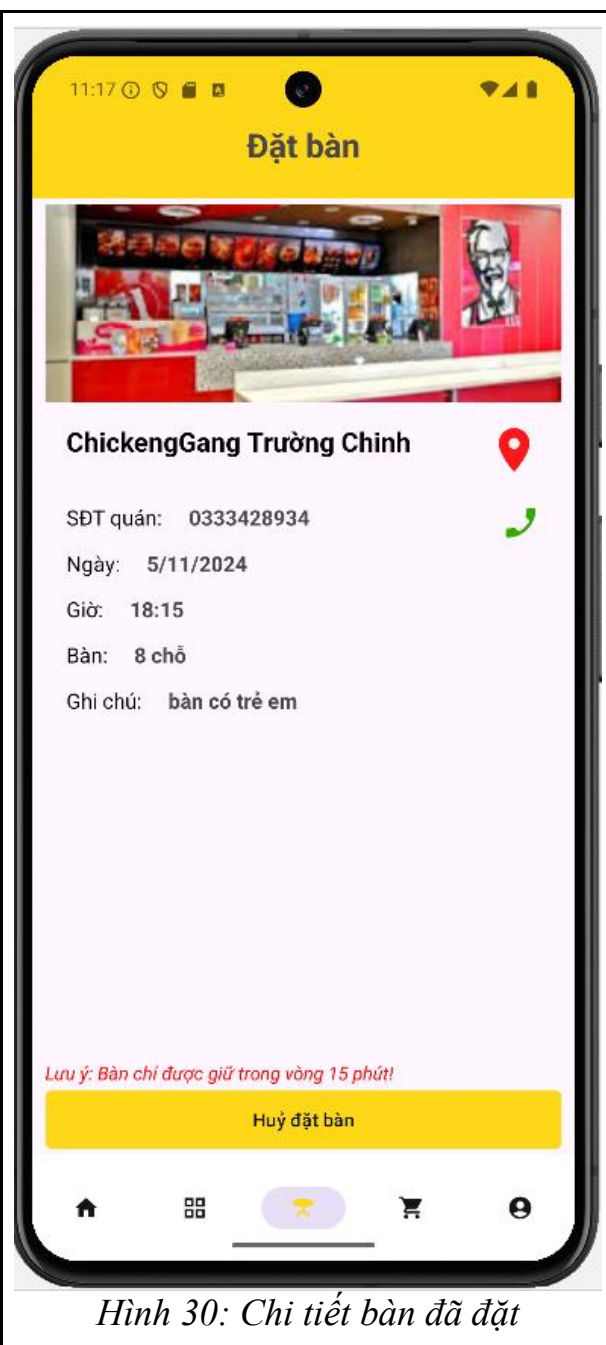
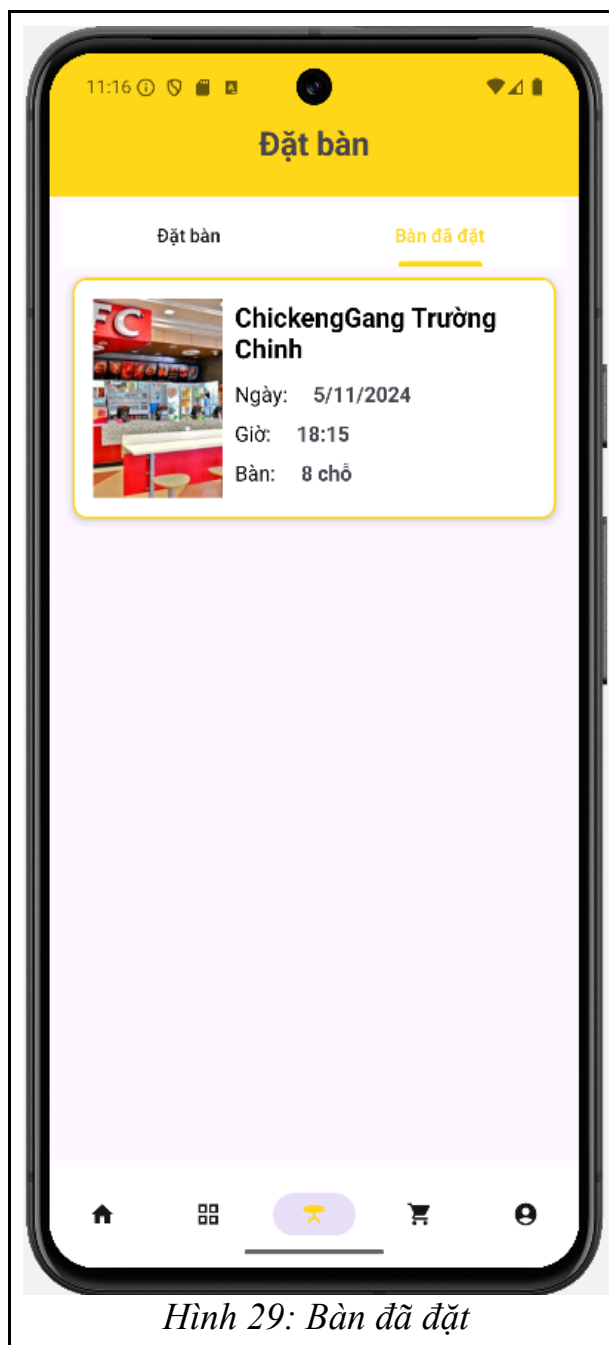
Hình 26: Danh mục

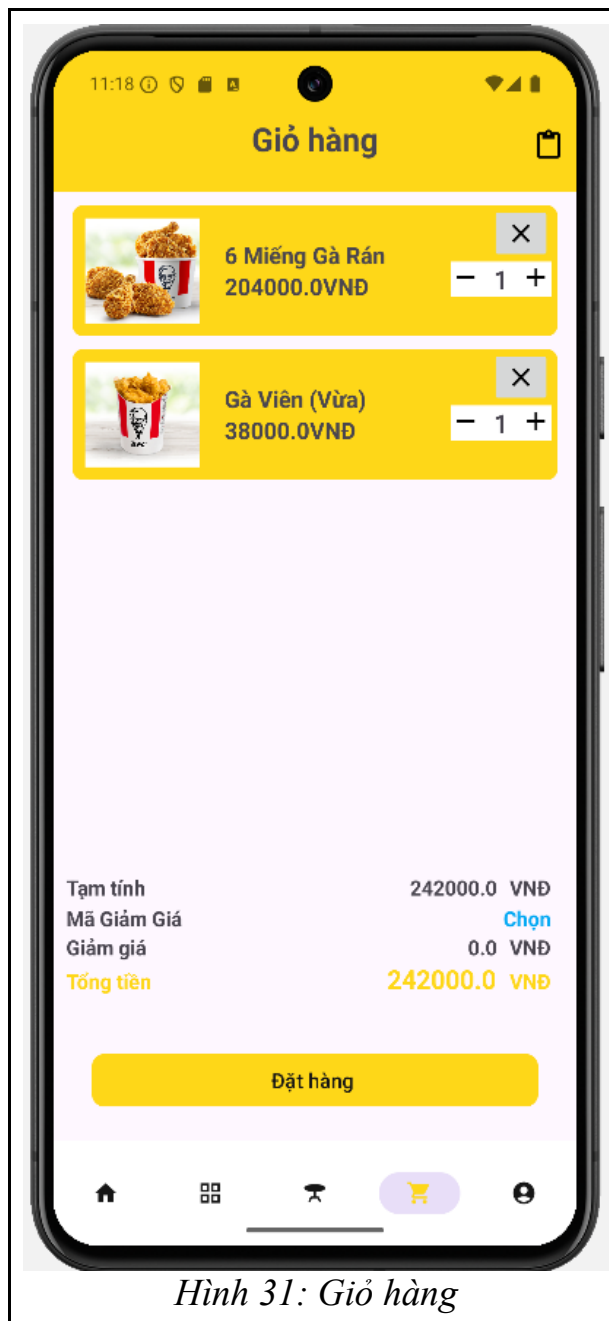


Hình 27: Chi tiết sản phẩm

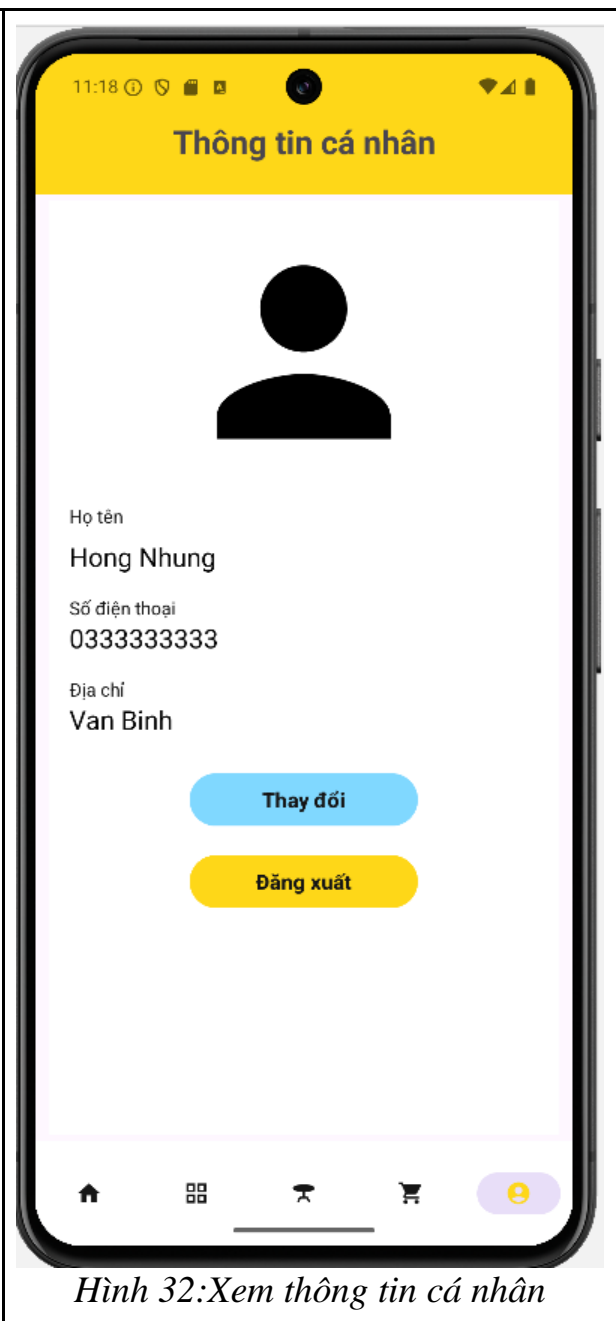


Hình 28: Đặt bàn

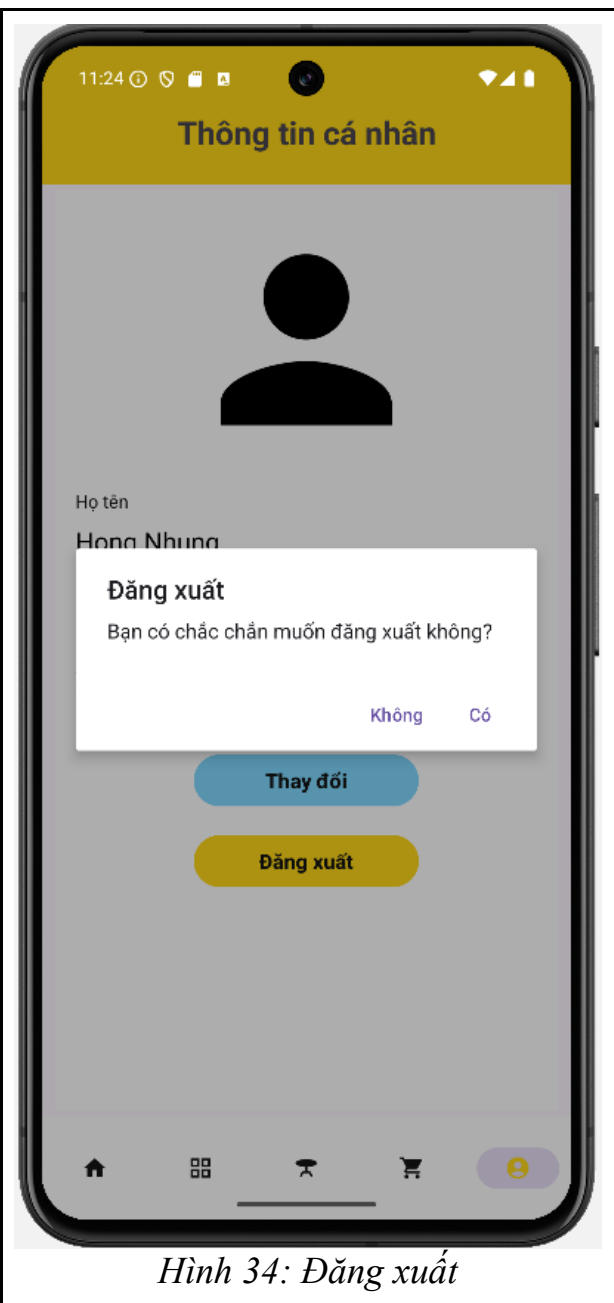
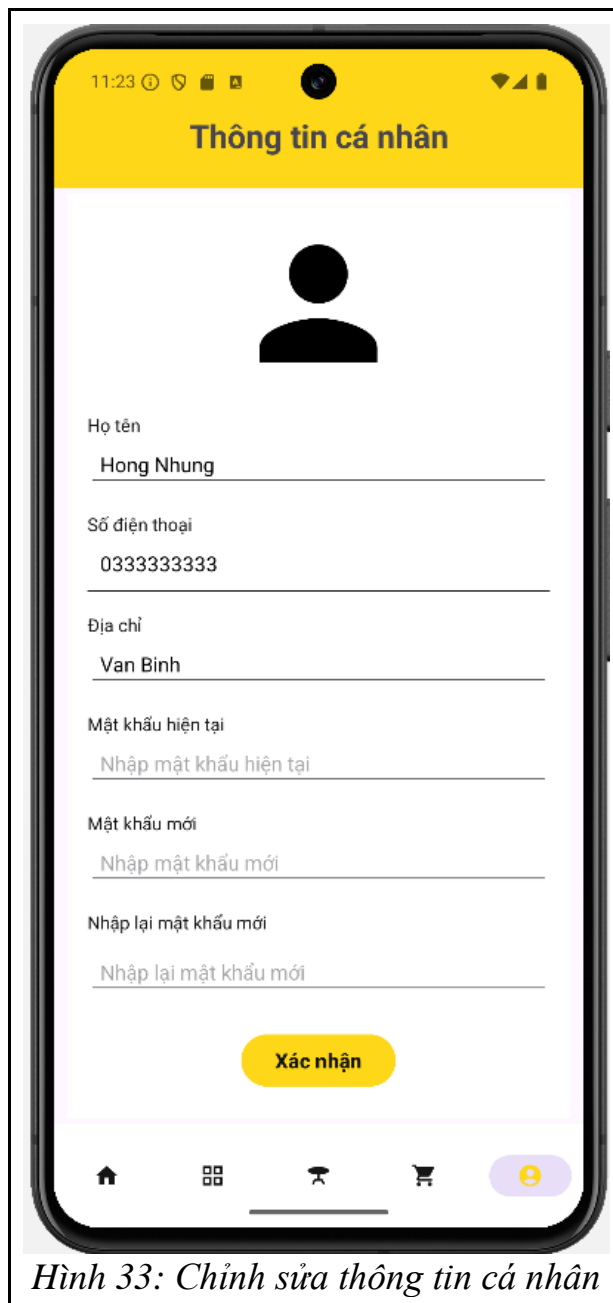




Hình 31: Giỏ hàng



Hình 32: Xem thông tin cá nhân



KẾT LUẬN

ChickenGang không chỉ mang lại sự tiện lợi trong việc đặt món ăn từ thực đơn đa dạng của mình, mà còn cung cấp mức giá ưu đãi hơn so với các nền tảng thương mại điện tử khác. Việc sử dụng ứng dụng này giúp giảm thiểu chi phí vận hành cho quán ăn, do không phụ thuộc vào các dịch vụ đặt đồ ăn online của bên thứ ba, vốn có thể làm tăng chi phí tiêu dùng.

● Ưu điểm :

- Tiện lợi và dễ sử dụng: Đăng ký, đăng nhập, chỉnh sửa thông tin cá nhân dễ dàng.
- Đa dạng lựa chọn món ăn: Hiện thị chi tiết món ăn, hỗ trợ lọc theo tên sản phẩm và tìm kiếm nhanh chóng.
- Quản lý đơn hàng nhanh chóng: Thêm món vào đơn và thanh toán dễ dàng, tạo trải nghiệm thuận tiện.
- Đặt chỗ dễ dàng: Đặt bàn trước và quản lý lịch đặt hiệu quả, tiết kiệm thời gian chờ đợi.
- Phù hợp cho nhiều nhóm khách hàng: Đáp ứng nhu cầu đa dạng của cá nhân, gia đình và nhóm bạn.

● Nhược điểm :

- Chưa hoàn thiện được tính năng đăng nhập bằng Google
- Chưa hoàn thiện được chức năng thanh toán online
- Hiệu năng chưa tối ưu: Khi ứng dụng phải xử lý nhiều dữ liệu hoặc hình ảnh chất lượng cao, dễ gặp tình trạng giật, lag, khiến trải nghiệm người dùng giảm.
- Thiếu tương thích trên nhiều thiết bị Android: Ứng dụng có thể chưa hoạt động mượt mà trên tất cả các loại thiết bị, dẫn đến lỗi hoặc hiệu suất không ổn định giữa các phiên bản và cấu hình máy khác nhau.
- Xử lý dữ liệu Firebase phức tạp: Khi nhiều người dùng thao tác cùng lúc, có thể xảy ra xung đột dữ liệu hoặc giảm tốc độ truy xuất. Firebase cũng gặp hạn chế khi thực hiện các truy vấn phức tạp, dễ làm chậm ứng dụng.

● Hướng phát triển chủ đề:

- Tối ưu hóa hiệu năng và tương thích đa nền tảng: Sử dụng các kỹ thuật tối ưu hình ảnh và dữ liệu để giảm tải ứng dụng, giúp ứng dụng hoạt động mượt mà hơn. Đảm bảo tương thích với nhiều loại thiết bị và các phiên bản Android, tập trung vào tối ưu hóa cho cả các thiết bị cấu hình thấp để mở rộng đối tượng người dùng.
- Nâng cao khả năng hoạt động offline: Triển khai tính năng cache dữ liệu quan trọng (menu, thông tin đặt hàng) để người dùng có thể sử dụng một số chức năng của ứng dụng khi không có kết nối mạng.
- Phát triển hệ thống gợi ý và cá nhân hóa: Sử dụng AI hoặc machine learning để gợi ý các món ăn dựa trên lịch sử đặt hàng, sở thích, giúp trải nghiệm của người dùng được cá nhân hóa hơn. Cung cấp các đề xuất khuyến mãi, combo phù hợp cho từng khách hàng để tăng tính tương tác và thúc đẩy doanh thu.
- Cải thiện quy trình kiểm thử : Áp dụng kiểm thử tự động (automated testing) để tăng hiệu quả kiểm thử và giảm thiểu lỗi phát sinh khi nâng cấp ứng dụng.
- Hỗ trợ đa ngôn ngữ: Phát triển ứng dụng với tính năng đa ngôn ngữ để phục vụ các nhóm khách hàng khác nhau, đặc biệt là nếu muốn mở rộng ra thị trường quốc tế.