ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

CÔNG NGHỆ PHẦN MỀM (CO3001)

BÀI TẬP LỚN: Đề tài

RESTAURANT POS 2.0

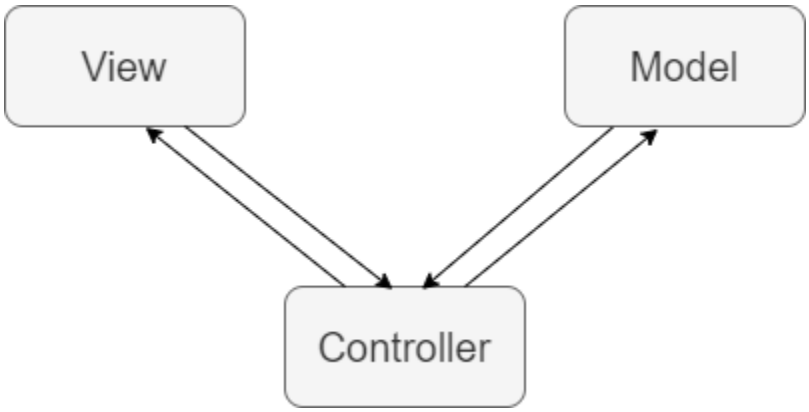
Giảng viên hướng dẫn:	**Lê Đình Thuận**	
Nhóm thực hiện:	**Nguyễn Hoài Thương**	**1912184**
	Nguyễn Hồng Dân	**1910916**
	Nguyễn Duy Uyên	**1912410**
	Trương Việt Hoàng	**1911207**
	Lê Mạnh Hùng	**1911283**
	Trương Gia Thịnh	**1915329**
	Trần Văn Thái	**1915121**

Tp. Hồ Chí Minh, Tháng 10/2021

Table of Contents

Task 3.1. Describe an architectural approach you will use to implement the desired system.

MVC (Model - View - Controller) là một mẫu thiết kế kiến trúc, nó quy định toàn bộ kiến trúc của ứng dụng.



1. Mô tả kiến trúc MVC

Nó có 3 thành phần chính, mỗi thành phần có một chức năng cụ thể riêng biệt:

a. Model

Đây là mức thấp nhất chịu trách nhiệm duy trì và xử lý dữ liệu một cách hợp lý. Bất kỳ thao tác lên dữ liệu như thêm hay truy xuất đều được thực hiện trong phần **Model**. Phần **Model** đáp ứng các yêu cầu của phần **Controller** bởi **Controller** không bao giờ tương tác trực tiếp với cơ sở dữ liệu. **Model** và cơ sở dữ liệu tương tác với nhau, sau đó **Model** cung cấp dữ liệu cần thiết cho **Controller**.

Model không bao giờ giao tiếp trực tiếp với phần **View**.

b. View

Việc biểu diễn dữ liệu được thực hiện bởi **View**. Nó chịu trách nhiệm tạo ra UI (user interface) cho người dùng. Vậy nên với một ứng dụng web, khi nhắc tới View thì ta sẽ nghĩ ngay tới phần html/css của trang web ấy. View được tạo ra bằng những dữ liệu thu thập được từ **Model**. Tuy nhiên, những dữ liệu này không được **Model** truyền thẳng cho View mà thông qua **Controller,** vậy nên nếu như **View** muốn có được dữ liệu để render thành UI cho người dùng thì phải giao tiếp với **Controller.**

c. Controller

Là một bộ điều khiển chính vì **Controller** là thành phần cho phép kết nối giữa **Model** và **View**, vì thế nó hoạt động như một trung gian để truyền dữ liệu. **Controller** không phải lo về việc xử lý các logic dữ liệu, nó chỉ nói cho **Model** biết cần làm cái gì. Sau khi nhận dữ liệu từ **Model**, nó xử lý và sau đó đem tất cả thông tin đó gửi đến cho **View** và giải thích cách biểu diễn đến người dùng.

View và **Model** không thể giao tiếp trực tiếp với nhau

2. Ưu điểm và nhược điểm của kiến trúc MVC

a. Ưu điểm

- Tách biệt **Model, View, Controller** với nhau.

- Các thành phần có thể tái sử dụng

- Dễ bảo trì

- Các thành phần của ứng dụng trong MVC có thể được triển khai và bảo trì độc lập.

- Kiến trúc MVC giúp kiểm thử các thành phần một cách độc lập.

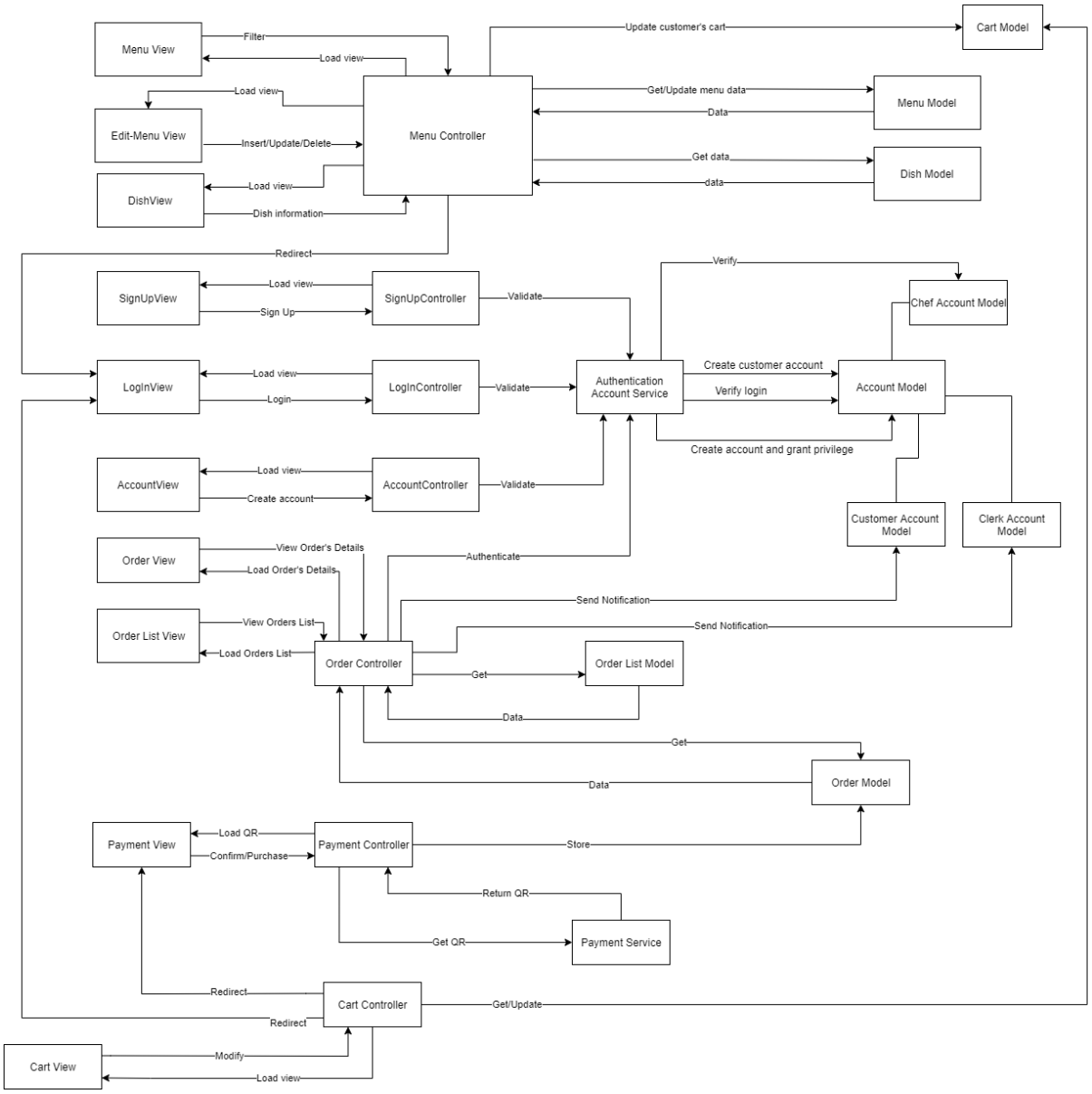
b. Nhược điểm

- Độ phức tạp cao,

- Không phù hợp với các ứng dụng nhỏ.

- Truy xuất dữ liệu không hiệu quả trong **View**.

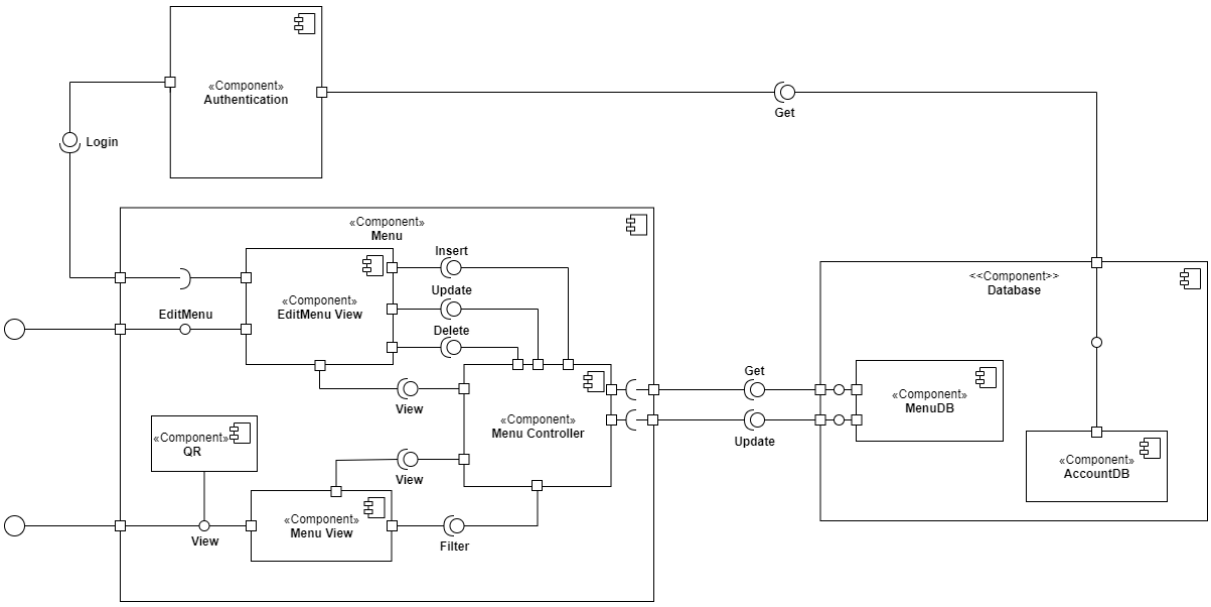
3. Kiến trúc MVC với hệ thống POS Restaurant



Link diagram: https://tinyurl.com/architectural-diagram

Task 3.2. Draw an implementation diagram for Major (not all) functional requirements.

1. Component diagram cho xem menu và edit menu



Mô tả diagram:

- Component Menu gồm 3 component chính là Authentication, Menu và Database. Trong component menu có các component là MenuEdit View, Menu controller, Menu View và QR code. Trong component database có 2 components con là MenuDB và AccountDB.

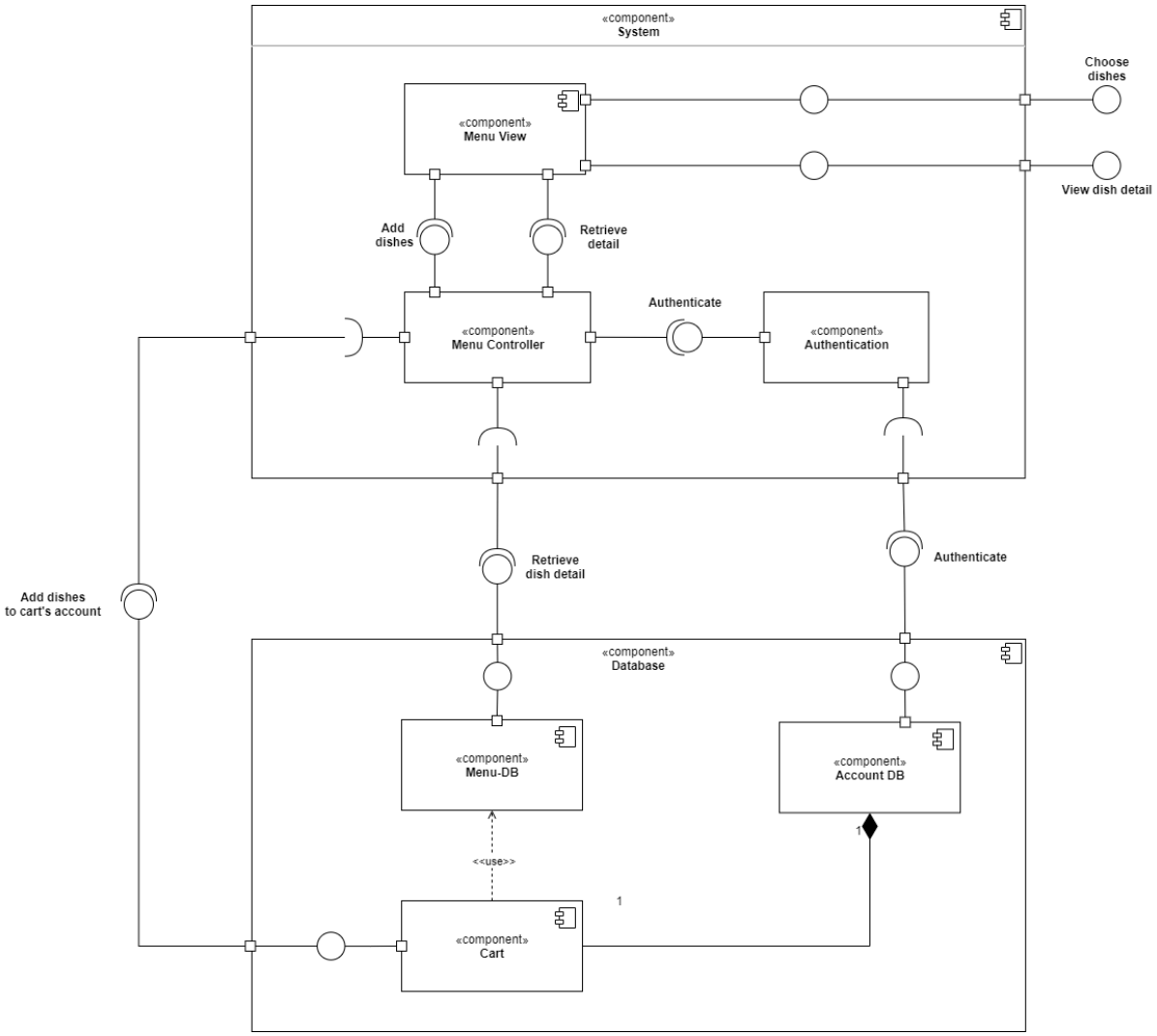
- Ở đây, menu cung cấp cho user khách hàng interface View và user quản lý interface EditMenu. Để thực hiện được các interface này thì các component sẽ cần yêu cầu các interface tương ứng phù hợp:

+ Interface View được cung cấp bởi component Menu View thông qua quét mã QR. Component Menu View yêu cầu các interface View (xem Menu), Filter (hiển thị menu theo phân loại món) từ component Menu Controller và Menu Controller sẽ yêu cầu các thao tác trên dữ liệu tương ứng phía dưới component Database thông qua Interface Get của component MenuDB.

+ Interface EditMenu được cung cấp bởi component MenuEdit View, component MenuEdit View cung cấp các Interface View (xem danh sách món), Insert (Thêm một món mới), Update (Cập nhật lại thông tin món), Delete (Xoá một món khỏi danh sách

Menu), các Interface này yêu cầu lấy dữ liệu qua Interface Get, và cập nhật dữ liệu qua Interface Update được cung cấp bởi component MenuDB. Để thực hiện được những thao tác này, User phải thực hiện đăng nhập với vai trò là quản lý thông qua interface Login được component MenuEdit View yêu cầu từ component Authentication. Component Authentication yêu cầu interface Get từ component AccountDB để xác thực việc đăng nhập.

2. Component diagram cho chọn món



Mô tả diagram:

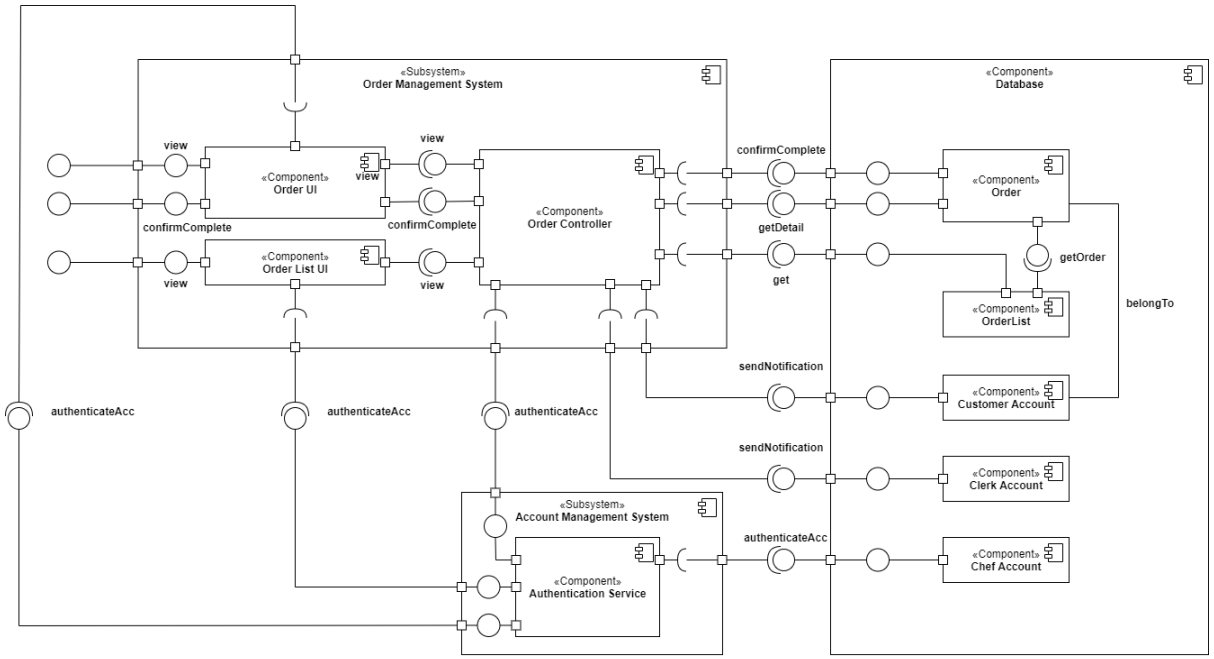
- Diagram gồm 3 component chính là Menu, Authentication và Database. Bên trong Menu, ta có 2 component con là MenuView và Menu Controller. Component cung cấp cho khách hàng 2 interface là chọn món (choose dishes) và xem thông tin chi tiết của món ăn.

- Khi sử dụng interface view dish detail (xem thông tin món ăn), khách hàng sẽ tương tác với component MenuView, component này sẽ tương tác với MenuController để lấy những thông tin chi tiết của món ăn mà nó đã load lên từ Database từ trước.

- Nếu sử dụng interface choose dishes (chọn món/thêm món vào giỏ), khách hàng cũng sẽ tương tác với MenuView và từ MenuView truyền thông điệp tới MenuController, tuy nhiên, MenuController sẽ yêu cầu dịch vụ của Component

authentication. Nếu khách hàng chưa đăng nhập thì component này sẽ bắt khách hàng phải đăng nhập hoặc tạo tài khoản mới. Nếu đã đăng nhập thì component này sẽ cập nhật dữ liệu về tài khoản (cụ thể là dữ liệu về giỏ hàng) trong Database.

3. Component diagram cho xem và xác nhận đơn hàng đã hoàn tất



Mô tả diagram:

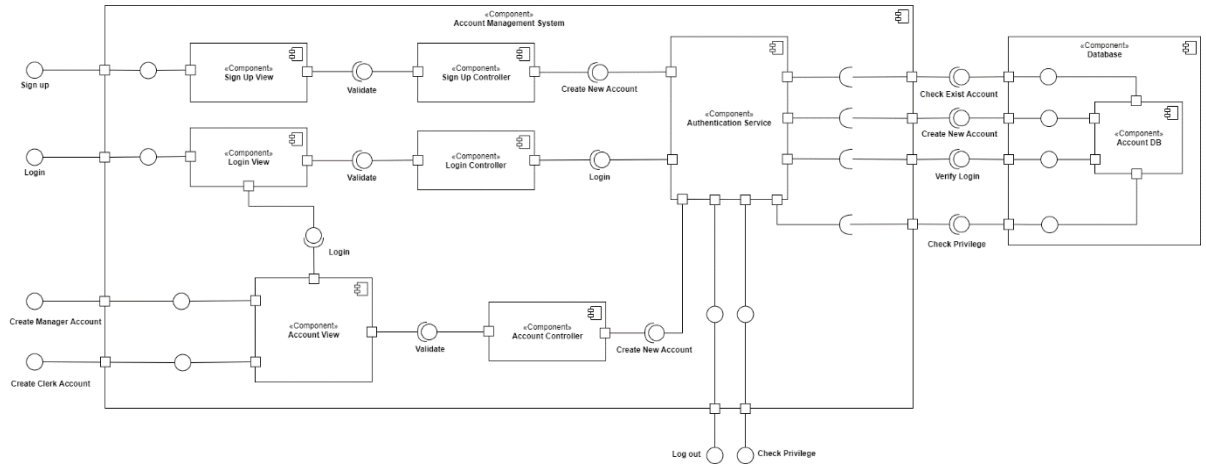
- Diagram gồm 3 thành phần chính: hệ thống quản lý đơn hàng, hệ thống xác thực tài khoản và database.

- Hệ thống quản lý đơn hàng bao gồm các component: Order UI yêu cầu giao diện thông tin chi tiết đơn hàng, Order List UI yêu cầu giao diện danh sách các đơn hàng chưa hoàn thành từ Order Controller. Sau đó, Order Controller sẽ tiếp tục yêu cầu đến Order DB và Order List DB.

- Để xác nhận hoàn tất đơn hàng, Order Controller yêu cầu giao diện đến hệ thống xác thực tài khoản để xác nhận mật khẩu của tài khoản đã đánh dấu hoàn tất đơn hàng. Hệ thống xác thực tài khoản sẽ yêu cầu đến Chef Account.

- Sau khi xác nhận, Order Controller gửi thông báo đến tài khoản của khách hàng đã đặt đơn hàng Customer Account và tài khoản của nhân viên phục vụ Clerk Account về thông tin hoàn tất.

4. Component diagram cho xác thực các loại tài khoản



Mô tả diagram:

Sign up (thông qua component Sign Up View), component Sign Up View sau đó yêu cầu giao diện Validate từ component Sign Up Controller, component Sign Up Controller sẽ yêu cầu giao diện Create New Account từ component Authentication Service, component Authentication Service sẽ tiếp tục yêu cầu các giao diện được cung cấp bởi component AccountDB (thuộc component Database) để hoàn thành đăng ký tài khoản cho khách hàng.

- Login (thông qua component Login View), component Login View sau đó yêu cầu giao diện Validate từ component Login Controller, component Login Controller sẽ yêu cầu giao diện Login từ component Authentication Service, component Authentication Service sẽ tiếp tục yêu cầu các giao diện được cung cấp bởi component AccountDB (thuộc component Database) để hoàn thành chức năng đăng nhập.

- Create Manager Account (thông qua component Account View), component Account View sau đó yêu cầu giao diện Validate từ component Account Controller, component Account Controller yêu cầu giao diện Create New Account từ component Authentication Service, component Authentication Service sẽ tiếp tục yêu cầu các giao diện được cung cấp bởi component AccountDB (thuộc component Database) để hoàn thành tạo tài khoản quản lý.

- Create Clerk Account (thông qua component Account View), component Account View sau đó yêu cầu giao diện Validate từ component Account Controller, component

Account Controller yêu cầu giao diện Create New Account từ component Authentication Service, component Authentication Service sẽ tiếp tục yêu cầu các giao diện được cung cấp bởi component AccountDB (thuộc component Database) để hoàn thành tạo tài khoản nhân viên.

- Logout (thông qua component Authenication Service) để đăng xuất khỏi hệ thống.

- Check Privilege (thông qua component Authenication Service), component Authenication Service sau đó yêu cầu giao diện Check Privilege từ component AccountDB (thuộc component Database) thực hiện chức năng kiểm tra các đặc quyền của tài khoản (sẽ được sử dụng bởi các component khác).

Component Database cung cấp các giao diện:

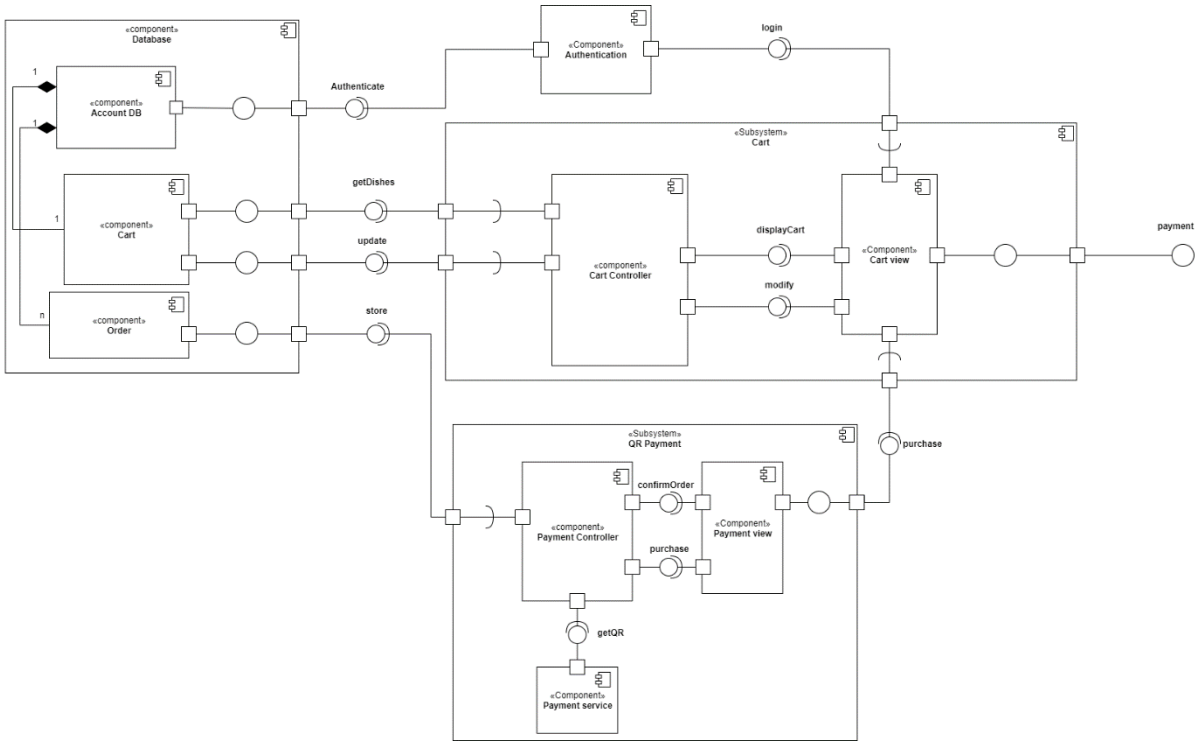
- Check Exist Account (thông qua component AccountDB): thực hiện kiểm tra tài khoản đã tồn tại hay không.

- Create New Account (thông qua component AccountDB): thực hiện lưu tài khoản mới tạo vào database.

- Verify Login (thông qua component AccountDB): thực hiện xác thực đăng nhập.

- Check privilege (thông qua component AccountDB): thực hiện kiểm tra các đặc quyền của tài khoản.

5. Component diagram cho xem giỏ hàng và thanh toán bằng ví điện tử



Mô tả diagram:

- Component Cart có thể điều hướng tới trang *login* bằng cách yêu cầu interface login từ component Authentication. Rồi component Authentication tiếp tục yêu cầu interface Authenticate được cung cấp bởi component Account DB (thuộc component Database) để hoàn thành chức năng đăng nhập.

- Component Cart cung cấp các giao diện để:

+ Hiển thị danh sách các món ăn trong giỏ hàng (thông qua component Cart view) component Cart view sau đó yêu cầu interface displayCart từ component Cart controller. Component Cart controller tiếp tục yêu cầu interface getDishes được cung cấp bởi component Cart (thuộc component Database) để hoàn thành công việc.

+ Cập nhật lại giỏ hàng sau khi đã thanh toán (thông qua component Cart view) component Cart view sau đó yêu cầu interface modify từ component Cart controller. Component Cart controller tiếp tục yêu cầu interface update được cung cấp bởi component Cart (thuộc component Database) để hoàn thành công việc.

- Component Cart có thể điều hướng tới trang *Payment* bằng cách yêu cầu interface purchase được cung cấp bởi component Payment view (thuộc component QR Payment). Component QR Payment cung cấp các giao diện để:

+ Xác thực đơn hàng (thông qua component Payment view) component Payment view yêu cầu interface từ component Payment controller.

+ Thanh toán đơn hàng (thông qua component Payment view) component Payment view yêu cầu interface purchase từ component Payment controller. Sau đó component Payment controller tiếp tục yêu cầu interface getQR được cung cấp bởi một dịch vụ ví điện từ (minh hoạ bởi Payment service) để lấy về đối tượng *mã QR* và hiển thị nó dưới dạng hình ảnh lên màn hình của trang thanh toán.

Sau khi thanh toán thành công thì component Payment controller yêu cầu interface store được cung cấp bởi component Order (thuộc component Database) để hoàn tất việc lưu trữ đơn hàng vào database.