+ Lớp src/main/java/controller/AuthenticationController.java

Coincidental Cohesion: Cohesion này xuất hiện khi các phương thức và thuộc tính trong lớp không có mối quan hệ logic nào với nhau và chỉ là một tập hợp các thành phần ngẫu nhiên. Trong lớp AuthenticationController, các phương thức và thuộc tính bao gồm:

Phương thức isAnonymousSession(): Kiểm tra xem phiên đăng nhập có là phiên ẩn danh không, thông qua việc gọi phương thức getMainUser() và xử lý ngoại lệ.

Phương thức getMainUser(): Trả về người dùng chính của phiên đăng nhập hiện tại, hoặc ném ra ngoại lệ ExpiredSessionException nếu phiên đã hết hạn.

Phương thức login(): Thực hiện quy trình đăng nhập cho người dùng thông qua việc gọi phương thức authenticate() từ UserDAO và mã hóa mật khẩu bằng phương thức md5().

Phương thức logout(): Thực hiện quy trình đăng xuất bằng cách đặt mainUser và expiredTime thành null.

Phương thức md5(): Mã hóa mật khẩu bằng thuật toán MD5.

Các phương thức và thuộc tính này không có mối quan hệ logic mạch lạc với nhau. Thay vào đó, chúng chỉ là một tập hợp các hành động và dữ liệu liên quan đến việc xác thực người dùng, nhưng không có mối quan hệ cụ thể nào giữa chúng.

+ Lớp src/main/java/controller/BaseController.java

Communicational Cohesion: Cohesion này xuất hiện khi các phương thức trong lớp được sử dụng cùng một tập hợp dữ liệu hoặc thông tin. Trong trường hợp này, cả hai phương thức checkMediaInCart() và getListCartMedia() đều liên quan đến việc quản lý giỏ hàng, đặc biệt là thông tin về các mặt hàng trong giỏ hàng.

Phương thức checkMediaInCart(Media media): Kiểm tra xem một phần tử media có tồn tại trong giỏ hàng không. Nó truy cập thông tin giỏ hàng thông qua SessionInformation.cartInstance để thực hiện kiểm tra.

Phương thức getListCartMedia(): Trả về danh sách các mặt hàng trong giỏ hàng. Nó cũng truy cập thông tin giỏ hàng thông qua SessionInformation.cartInstance.

Cả hai phương thức này đều liên quan chặt chẽ đến việc quản lý thông tin trong giỏ hàng, và chúng chia sẻ cùng một tập hợp dữ liệu - đó là thông tin giỏ hàng được lưu trữ trong SessionInformation.cartInstance.

+ Lớp src/main/java/controller/PaymentController.java

Communicational Cohesion: Cohesion này xuất hiện khi các phương thức trong lớp được sử dụng cùng một tập hợp dữ liệu hoặc thông tin. Trong trường hợp này, các phương thức trong lớp PaymentController đều liên quan chặt chẽ đến quá trình thanh toán và quản lý thông tin thanh toán:

Phương thức getExpirationDate(String date): Xác nhận và trả về ngày hết hạn của thẻ tín dụng, và truy cập các thông tin liên quan đến thẻ tín dụng như thẻ số, tên chủ thẻ, ngày hết hạn, mã bảo mật.

Phương thức payOrder(int amount, String contents, String cardNumber, String cardHolderName, String expirationDate, String securityCode): Thực hiện quá trình thanh toán, truy cập thông tin thẻ tín dụng và thực hiện giao dịch thanh toán thông qua giao diện InterbankInterface.

Phương thức emptyCart(): Xóa toàn bộ mục hàng khỏi giỏ hàng. Mặc dù phương thức này có vẻ không liên quan trực tiếp đến thanh toán, nhưng nó vẫn liên quan đến quản lý các thông tin đơn hàng và giỏ hàng, nằm trong phạm vi của các chức năng liên quan đến thanh toán.

Cả ba phương thức này đều liên quan chặt chẽ đến quá trình thanh toán và quản lý thông tin thanh toán, và chúng chia sẻ cùng một tập hợp dữ liệu và thông tin về giao dịch thanh toán.

+Lớp src/main/java/controller/ViewCartController.java

Functional Cohesion: Đây là mức độ cohesion tốt nhất trong đó các phương thức trong lớp phục vụ cho một mục đích cụ thể hoặc hoàn thành một nhiệm vụ cụ thể. Trong trường hợp này, cả hai phương thức trong lớp ViewCartController đều liên quan chặt chẽ đến việc hiển thị và quản lý thông tin giỏ hàng.

Phương thức checkAvailabilityOfProduct(): Thực hiện kiểm tra sự có sẵn của các sản phẩm trong giỏ hàng. Nó là một phần của quá trình hiển thị giỏ hàng và cung cấp thông tin chi tiết về trạng thái của các sản phẩm trong giỏ hàng.

Phương thức getCartSubtotal(): Tính tổng tiền của giỏ hàng. Đây là một phần của quá trình hiển thị giỏ hàng và cung cấp thông tin về tổng tiền của giỏ hàng.

Cả hai phương thức này đều phục vụ cho mục đích cụ thể là hiển thị và quản lý thông tin giỏ hàng. Chúng không chỉ chia sẻ cùng một tập hợp dữ liệu hoặc thông tin, mà còn hoàn thành một nhiệm vụ cụ thể - đó là liên quan đến việc hiển thị giỏ hàng.

+ Lớp src/main/java/dao/media/MediaDAO.java

Functional Cohesion: Đây là mức độ cohesion tốt nhất trong đó các phương thức trong lớp phục vụ cho một mục đích cụ thể hoặc hoàn thành một nhiệm vụ cụ thể. Trong trường hợp này, các phương thức trong lớp MediaDAO đều liên quan chặt chẽ đến việc truy cập và quản lý dữ liệu liên quan đến các phương tiện (media) trong cơ sở dữ liệu:

Phương thức getAllMedia(): Lấy tất cả các phương tiện từ cơ sở dữ liệu. Nhiệm vụ của phương thức này là truy xuất dữ liệu về tất cả các phương tiện từ cơ sở dữ liệu và trả về danh sách chúng.

Phương thức getMediaById(int id): Lấy thông tin của một phương tiện dựa trên id. Nhiệm vụ của phương thức này là truy xuất thông tin chi tiết của một phương tiện dựa trên id được cung cấp và trả về đối tượng Media tương ứng.

Phương thức updateMediaFieldById(String tbname, int id, String field, Object value): Cập nhật một trường dữ liệu của một phương tiện dựa trên id. Nhiệm vụ của phương thức này là cập nhật một trường dữ liệu cụ thể của một phương tiện dựa trên id và giá trị được cung cấp.

+ Lớp src/main/java/entity/cart/Cart.java

Functional Cohesion: Đây là mức độ cohesion tốt nhất trong đó các phương thức trong lớp phục vụ cho một mục đích cụ thể hoặc hoàn thành một nhiệm vụ cụ

thể. Trong trường hợp này, tất cả các phương thức trong lớp Cart đều liên quan chặt chẽ đến việc quản lý giỏ hàng và thực hiện các tác vụ liên quan đến giỏ hàng.

Phương thức addCartMedia(CartItem cm): Thêm một mục hàng vào giỏ hàng.

Phương thức removeCartMedia(CartItem cm): Xóa một mục hàng khỏi giỏ hàng.

Phương thức getListMedia(): Trả về danh sách các mục hàng trong giỏ hàng.

Phương thức emptyCart(): Xóa tất cả các mục hàng khỏi giỏ hàng.

Phương thức getTotalMedia(): Tính tổng số lượng các mục hàng trong giỏ hàng.

Phương thức calSubtotal(): Tính tổng giá trị của các mục hàng trong giỏ hàng.

Phương thức checkAvailabilityOfProduct(): Kiểm tra sự có sẵn của các sản phẩm trong giỏ hàng.

Phương thức checkMediaInCart(Media media): Kiểm tra xem một sản phẩm đã có trong giỏ hàng chưa.

Tất cả các phương thức này đều phục vụ cho mục đích cụ thể là quản lý giỏ hàng và hoàn thành các nhiệm vụ cụ thể liên quan đến giỏ hàng. Chúng không chỉ chia sẻ cùng một tập hợp dữ liệu hoặc thông tin, mà còn hoàn thành một nhiệm vụ cụ thể và liên quan chặt chẽ với nhau.

+ Lớp src/main/java/entity/cart/CartItem.java

Class CartItem có functional cohesion vì các phương thức và thuộc tính đều liên quan chặt chẽ đến việc quản lý thông tin về mặt hàng trong giỏ hàng.

Thuộc tính:

private Media media: Đại diện cho một đối tượng Media (một mặt hàng trong giỏ hàng).

private int quantity: Lưu trữ số lượng của mặt hàng trong giỏ hàng.

private int price: Lưu trữ giá của mặt hàng.

Phương thức khởi tạo:

public CartItem(Media media, Cart cart, int quantity, int price): Phương thức này được sử dụng để tạo mới một đối tượng CartItem. Nó nhận vào thông tin về Media, số lượng, và giá của mặt hàng.

public CartItem(): Phương thức khởi tạo mặc định không có tham số.

Các phương thức khác:

public Media getMedia(): Trả về đối tượng Media của CartItem.

public void setMedia(Media media): Thiết lập đối tượng Media cho CartItem.

public int getQuantity(): Trả về số lượng của mặt hàng.

public void setQuantity(int quantity): Thiết lập số lượng cho mặt hàng.

public int getPrice(): Trả về giá của mặt hàng.

public void setPrice(int price): Thiết lập giá cho mặt hàng.

@Override public String toString(): Ghi đè phương thức toString() để trả về một chuỗi biểu diễn của CartItem. Chuỗi này bao gồm thông tin về media và quantity.

+ Lớp src/main/java/entity/invoice/Invoice.java

functional cohesion

Class Invoice có cohesion vì các phương thức và thuộc tính đều liên quan chặt chẽ đến việc quản lý thông tin về hóa đơn.

+ Lớp src/main/java/subsystem/InterbankSubsystem.java

Functional Cohesion:

Functional Cohesion xảy ra khi các phương thức trong một class hoặc module liên quan chặt chẽ đến cùng một nhiệm vụ hoặc chức năng cụ thể.

Trong trường hợp của InterbankSubsystem, các phương thức đều liên quan đến việc giao tiếp với hệ thống Interbank để thực hiện các giao dịch.

+ Lớp src/main/java/subsystem/interbank/InterbankPayloadConverter.java

Phương thức convertToRequestPayload(): Chuyển đổi thông tin thanh toán từ các thực thể nội bộ thành định dạng yêu cầu của Interbank. Phương thức này thực hiện hai nhiệm vụ: chuyển đổi dữ liệu và xây dựng một payload yêu cầu hợp lệ để gửi đến Interbank.

Phương thức extractPaymentTransaction(): Phân tích phản hồi từ máy chủ Interbank và chuyển đổi nó thành một giao dịch thanh toán hợp lệ. Phương thức này thực hiện các bước để trích xuất thông tin giao dịch từ phản hồi và xây dựng một đối tượng PaymentTransaction từ nó.

Phương thức convertJSONResponse(): Chuyển đổi phản hồi từ máy chủ Interbank từ chuỗi JSON thành một Map chứa dữ liệu. Phương thức này chỉ có mối quan hệ gián tiếp với các phương thức khác trong lớp, không có mối quan hệ trực tiếp.

Phương thức getToday(): Trả về thời gian hiện tại dưới dạng một chuỗi theo định dạng cụ thể. Phương thức này không có mối quan hệ trực tiếp với các phương thức khác trong lớp.

Mặc dù các phương thức trong lớp này đều liên quan đến việc xử lý dữ liệu liên quan đến giao dịch thanh toán, nhưng không có mối quan hệ chặt chẽ giữa các phương thức và chúng thực hiện các nhiệm vụ không liên quan đến nhau. Do đó, lớp InterbankPayloadConverter có mức độ cohesion là "coincidental cohesion".

+ Lớp src/main/java/views/screen/shipping/ShippingScreenHandler.java

Thuộc tính và phương thức để thao tác với các thành phần giao diện người dùng như nhãn, ô nhập và hộp combo để thu thập thông tin vận chuyển từ người dùng.

Phương thức submitDeliveryInfo(): Xử lý sự kiện khi người dùng nhấn nút để gửi thông tin vận chuyển. Phương thức này gọi các phương thức khác để kiểm tra và xử lý thông tin vận chuyển, sau đó tạo màn hình hóa đơn và hiển thị nó.

Phương thức preprocessDeliveryInfo(): Xử lý thông tin vận chuyển trước khi gửi nó đi, bao gồm kiểm tra tính hợp lệ và thiết lập nó cho đối tượng đơn hàng.

Phương thức getBController(): Trả về một tham chiếu đến đối tượng PlaceOrderController để xử lý các tác vụ liên quan đến đặt hàng.

Tất cả các thành phần trong lớp này đều liên quan chặt chẽ với việc thu thập và xử lý thông tin vận chuyển, cùng hỗ trợ chức năng hiển thị màn hình hóa đơn và gửi thông tin vận chuyển. Do đó, lớp ShippingScreenHandler có mức độ cohesion là "logical cohesion".