Vi phạm SRP

src\main\java\controller\AuthenticationController.java

```
public boolean isAnonymousSession() {
    try {
        getMainUser();
        return false;
    } catch (Exception ex) {
        return true;
    }
}
```

Giải thích: Phương thức isAnonymousSession() không chỉ có trách nhiệm kiểm tra xem phiên làm việc hiện tại có phải là phiên của một người dùng ẩn danh hay không, mà còn thực hiện việc lấy thông tin người dùng chính thức nếu phiên làm việc không ẩn danh. Điều này làm cho phương thức này có nhiều trách nhiệm, không chỉ kiểm tra trạng thái phiên mà còn thực hiện hành động khác.

```
public User getMainUser() throws ExpiredSessionException {
    if (SessionInformation.mainUser == null || SessionInformation.expiredTime
== null || SessionInformation.expiredTime.isBefore(LocalDateTime.now())) {
        logout();
        throw new ExpiredSessionException();
    } else return SessionInformation.mainUser.cloneInformation();
}
```

Giải thích: Phương thức getMainUser() không chỉ trả về thông tin người dùng chính thức, mà còn kiểm tra xem phiên làm việc có hết hạn hay không. Điều này làm cho phương thức này có nhiều trách nhiệm và không tuân thủ nguyên lý SRP.

```
public void login(String email, String password) throws Exception {
    try {
        User user = new UserDAO().authenticate(email, md5(password));
        if (Objects.isNull(user)) throw new FailLoginException();
        SessionInformation.mainUser = user;
        SessionInformation.expiredTime = LocalDateTime.now().plusHours(24);
    } catch (SQLException ex) {
        throw new FailLoginException();
    }
}
```

Giải thích: Phương thức login() thực hiện quá nhiều công việc, bao gồm xác thực người dùng, tạo phiên làm việc mới và xử lý các ngoại lệ. Điều này làm cho phương thức này không tách biệt các trách nhiệm và vi phạm nguyên lý SRP.

Vi phạm SRP

src\main\java\controller\BaseController.java

```
public CartItem checkMediaInCart(Media media){
    return SessionInformation.cartInstance.checkMediaInCart(media);
}
```

Giải thích: Phương thức checkMediaInCart(Media media) trong BaseController không chỉ có trách nhiệm kiểm tra xem một phần tử Media có trong giỏ hàng hay không, mà còn trả về CartItem tương ứng nếu tìm thấy. Điều này làm cho phương thức này có nhiều trách nhiệm, không chỉ thực hiện việc kiểm tra mà còn trả về dữ liệu liên quan.

```
public List getListCartMedia(){
    return SessionInformation.cartInstance.getListMedia();
}
```

Giải thích: Phương thức getListCartMedia() trong BaseController không chỉ có trách nhiệm trả về danh sách các mục trong giỏ hàng, mà còn trả về một danh sách không rõ ràng kiểu dữ liệu (raw List). Điều này làm cho phương thức này có nhiều trách nhiệm và không tuân thủ nguyên lý SRP.

Vi phạm SRP

src\main\java\controller\PaymentController.java

```
public Map<String, String> payOrder(int amount, String contents, String cardNumber, String cardHolderName,
        String expirationDate, String securityCode) {
    Map<String, String> result = new Hashtable<String, String>();
    result.put("RESULT", "PAYMENT FAILED!");
    try {
        this.card = new CreditCard(
                        cardNumber,
                        cardHolderName,
                        getExpirationDate(expirationDate),
                        Integer.parseInt(securityCode));

        this.interbank = new InterbankSubsystem();
        PaymentTransaction transaction = interbank.payOrder(card, amount, contents);

        result.put("RESULT", "PAYMENT SUCCESSFUL!");
        result.put("MESSAGE", "You have successfully paid the order!");
    } catch (PaymentException | UnrecognizedException ex) {
        result.put("MESSAGE", ex.getMessage());
    }
```

```
  return result;
}
```

-Giải thích: Phương thức `payOrder` không chỉ có trách nhiệm thực hiện quá trình thanh toán (`payOrder`), mà còn có trách nhiệm khởi tạo `CreditCard` từ các thông tin đầu vào và xử lý các ngoại lệ. Điều này làm cho phương thức này có nhiều trách nhiệm, không tuân thủ nguyên lý SRP.

Vi phạm SRP

src\main\java\controller\PlaceOrderController.java

```
public void validateDeliveryInfo(HashMap<String, String> info) throws InterruptedException, IOException, InvalidDeliveryInfoException {
  if (validatePhoneNumber(info.get("phone"))
  || validateName(info.get("name"))
  || validateAddress(info.get("address"))) return;
  else throw new InvalidDeliveryInfoException();
}

public boolean validatePhoneNumber(String phoneNumber) {
if (phoneNumber.length() != 10) return false;
if (!phoneNumber.startsWith("0")) return false;
try {
Integer.parseInt(phoneNumber);
} catch (NumberFormatException e) {
return false;
}
return true;
}
```

```
public boolean validateName(String name) {
if (Objects.isNull(name)) return false;
String patternString = "^[a-zA-Z\s]*$";
Pattern pattern = Pattern.compile(patternString);
Matcher matcher = pattern.matcher(name);
return matcher.matches();
}

public boolean validateAddress(String address) {
if (Objects.isNull(address)) return false;
String patternString = "^[a-zA-Z\s]*$";
Pattern pattern = Pattern.compile(patternString);
Matcher matcher = pattern.matcher(address);
return matcher.matches();
}
```

Giải thích: Phương thức `validateDeliveryInfo` không chỉ có trách nhiệm kiểm tra thông tin giao hàng (`DeliveryInfo`), mà còn có trách nhiệm xử lý và kiểm tra các điều kiện của các trường dữ liệu riêng lẻ như số điện thoại, tên, địa chỉ. Điều này làm cho phương thức này có nhiều trách nhiệm và không tuân thủ nguyên lý SRP.

Vi phạm nguyên lý SRP

src\main\java\entity\cart\Cart.java

```
public class Cart {

    private List<CartItem> lstCartItem;

    // Phương thức addCartMedia thêm một mặt hàng vào giỏ hàng
```

```
public void addCartMedia(CartItem cm){
    lstCartItem.add(cm);
}

// Phương thức removeCartMedia xóa một mặt hàng khỏi giỏ hàng
public void removeCartMedia(CartItem cm){
    lstCartItem.remove(cm);
}

// Phương thức getListMedia trả về danh sách các mặt hàng trong giỏ hàng
public List getListMedia(){
    return lstCartItem;
}

// Phương thức emptyCart xóa tất cả các mặt hàng trong giỏ hàng
public void emptyCart(){
    lstCartItem.clear();
}

// Phương thức getTotalMedia tính tổng số lượng mặt hàng trong giỏ hàng
public int getTotalMedia(){
    int total = 0;
    for (Object obj : lstCartItem) {
        CartItem cm = (CartItem) obj;
        total += cm.getQuantity();
    }
    return total;
```

```
    }

    // Phương thức calSubtotal tính tổng tiền của các mặt hàng trong giỏ hàng
    public int calSubtotal(){
        int total = 0;
        for (Object obj : lstCartItem) {
            CartItem cm = (CartItem) obj;
            total += cm.getPrice()*cm.getQuantity();
        }
        return total;
    }

    // Phương thức checkAvailabilityOfProduct kiểm tra tính khả dụng của các sản phẩm trong giỏ hàng
    public void checkAvailabilityOfProduct() throws SQLException{
        boolean allAvailable = true;
        for (Object object : lstCartItem) {
            CartItem cartItem = (CartItem) object;
            int requiredQuantity = cartItem.getQuantity();
            int availQuantity = cartItem.getMedia().getQuantity();
            if (requiredQuantity > availQuantity) allAvailable = false;
        }
        if (!allAvailable) throw new MediaNotAvailableException("Some media not available");
    }

    // Phương thức checkMediaInCart kiểm tra xem một sản phẩm có trong giỏ hàng hay không
```

```
    public CartItem checkMediaInCart(Media media){
        for (CartItem cartItem : lstCartItem) {
            if (cartItem.getMedia().getId() == media.getId()) return cartItem;
        }
        return null;
    }
}
```

Lí do vi phạm SRP:

Lớp Cart chịu trách nhiệm quản lý danh sách các mặt hàng trong giỏ hàng, tính toán tổng số lượng và tổng tiền của các mặt hàng, kiểm tra tính khả dụng của các sản phẩm và kiểm tra xem một sản phẩm có trong giỏ hàng hay không. Điều này làm cho lớp này có quá nhiều trách nhiệm và không tuân thủ nguyên lý SRP.

Vi phạm nguyên lý OCP

src\main\java\entity\cart\Cart.java

```
public class Cart {


    // Các phương thức của lớp Cart thực hiện trực tiếp các chức năng cụ thể như thêm, xóa mặt hàng, tính tổng tiền, kiểm tra tính khả dụng của sản phẩm, ...
    // Không dễ dàng mở rộng để thêm hoặc thay đổi các chức năng mới mà không cần phải sửa đổi mã nguồn hiện tại.
}
```

Lí do vi phạm OCP:

Lớp Cart không dễ dàng mở rộng để thêm hoặc thay đổi các chức năng mới mà không cần phải sửa đổi mã nguồn hiện tại. Điều này làm giảm tính linh hoạt và tái sử dụng của mã nguồn, không tuân thủ nguyên lý OCP.