**Coupling in Payment**

1. **Controller package - PayOrderController**

A screen shot of a computer code

Description automatically generated

PayOrderController uses UnsupportedEncodingException, This is very small and does not affect system independence

* External coupling

A screen shot of a computer code

Description automatically generated

payOrder(Order order) receives an Order object as a parameter, which creates stamp coupling between PayOrderController and Order.

A close-up of a computer screen

Description automatically generated

onUpdateSuccess(PaymentTransaction trans) receives a PaymentTransaction object, creating a stamp coupling between PayOrderController and PaymentTransaction.

* Stamp coupling



subsystem.payOrder(order.getTotalCost(), "Pay AIMS invoice") passes the underlying data (real number and character string) to IPayment's payOrder method, creating data coupling.

* Data coupling

1. **Controller package- PlaceOrderController**

IOException in the placeOrder method.

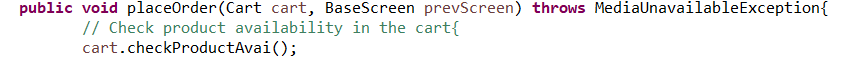
A screenshot of a computer program

Description automatically generated

IOException is an exception that is beyond the control of PlaceOrderController

* **External coupling**

Cart, CartMedia, Order, DeliveryInfo are used in the methods of PlaceOrderController.

  A screen shot of a computer code

Description automatically generated

PlaceOrderController uses these objects to perform product availability checks, generate invoices, and validate deliveries.

* **data coupling.**

1. **Entity-payment- PaymentTransaction**

This class focuses mainly on storing payment transaction data.

The PaymentTransaction class does not have the content coupling, common coupling, control coupling, stamp coupling, and message coupling types.

The only **external coupling** is due to using class Date of JAVA jdk

=> This is a layer with a low level of coupling

1. **subsystem package**
2. VNPayController class

This is class with a relatively low level of coupling and focuses mainly on payment processing with VNPay.

VNPayController calls the onUpdateSuccess and onUpdateFailure methods of the client (PayOrderController).

A screen shot of a computer code

Description automatically generatedVNPayController is controlling the execution flow by calling client methods. It make control coupling

Request and Response are used to create and process transaction requests. A screenshot of a computer code

Description automatically generated

VNPayController uses Request and Response objects to perform transaction-related operations,

* **data coupling.**

1. Request class

This is class with a low level of coupling and focuses mainly on creating payment requests for VNPay.

The Request class does not have content coupling, common coupling, control coupling, stamp coupling, and message coupling.

This class uses classes and packages from the Java SDK

* **external coupling**

1. Respone class

Response is controlling the execution flow by processing the response from VNPay and deciding to process the payment transaction accordingly.

A screen shot of a computer code

Description automatically generatedResponse receives a processResponse method to process the response from VNPay and decide to process the payment transaction accordingly based on the received error code

#### => Control Coupling

1. VNPayScreen class

The VNPayScreen class calls the processResponse method of VNPayController to process the response from VNPay.

A screen shot of a computer code

Description automatically generatedVNPayScreen controls the execution flow by calling VNPayController's method to process the response from VNPay.

* Control coupling

1. VNPay config class

vnp\_ReturnUrl, vnp\_TmnCode, secretKey, vnp\_Version, vnp\_Command are used to configure parameters for VNPay.

A screenshot of a computer code

Description automatically generatedThe VNPayConfig class uses these variables to configure parameters for VNPay,

* data coupling.