# 5. BÀI THỰC HÀNH SỐ 03 – THIẾT KẾ LỚP VÀ MÔ HÌNH HÓA DỮ LIỆU

Họ và tên: Nguyễn Quang Trung - 20180188

## 5.1. MỤC ĐÍCH VÀ NỘI DUNG

Trong bài thực hành này, chúng ta sẽ làm quen với 02 bước còn lại trong thiết kế chi tiết (detailed design process), gồm có thiết kế lớp (class design) và mô hình hóa dữ liệu (data modeling). Kết thúc bài thực hành, người học có thể đạt được đủ kỹ nâng để hoàn thiện Software Design Document (SDD).

## 5.2. CHUẨN BỊ

Người học cần tự hoàn thiện trước buổi học: bước thiết kế kiến trúc của phần mềm (Architectural Design) và thiết kế giao diện của bước đầu thiết kế chi tiết. Kết quả của bước thiết kế kiến trúc phần mềm sẽ là đầu vào cho bước thiết kế chi tiết. Đồng thời kết quả của thiết kế giao diện sẽ cần thiết để hoàn thiện thiết kế chi tiết.

## 5.3. NỘI DUNG CHI TIẾT

### 5.3.1. Thiết kế lớp (Class Design)

#### 5.3.1.1. Bước đầu tạo các lớp thiết kế

Trong phần này, chúng ta sẽ ánh xạ các thành phần thiết kế (design elements, ví dụ: lớp – class, nhóm các lớp – group of classes, gói – package, subsystem) từ các lớp phân tích (analysis classes). Mỗi lớp thiết kế nên chỉ phục vụ tốt một mục đích duy nhất. Chúng ta sẽ xác định các lớp thiết kế dựa vào biểu đồ lớp kiến trúc và khuôn mẫu (stereotype) của lớp đấy. Lưu ý rằng chúng ta chưa ứng dụng các mẫu thiết kế (design patterns) trong bài thực hành này.

##### Thiết kế lớp boundary

###### Lớp boundary: Giao diện người dùng (User interface)

Trong Case study, chúng ta sử dụng JavaFX để xây dựng giao diện. Do đó, từ góc nhìn kiến trúc, mỗi lớp boundary giao diện người dùng tương đương với lớp thiết kế phụ trách xử lý sự kiện hoặc hành động của người dùng (được bắt bởi FXML tương ứng). Trong JavaFX, mặc dù các lớp thiết kế này thường được gọi là “controller” của các tệp FXML, chúng không thực sự đóng vai trò như lớp control trong UML. Do vậy, đa số các lớp xử lý sự kiện hiện tại đã khá đơn giản, và ánh xạ là 1-1.

###### Lớp boundary: Giao diện hệ thống (System/device boundary)

Trong bài thực hành trước, chúng ta đã “evolve”/ánh xạ lớp boundary cho liên ngân hàng Interbank thành một subsystem. Tuy nhiên, subsystem này thiết kế chưa được tốt: InterbankSubsystemController quá phức tạp, đồng thời một phần InterbankBoundary vẫn có thể tái sử dụng. Hiện tại trong mô tả của Case Study, chỉ có nhắc đến một hệ thống thông tin web bên ngoài (Interbank), nhưng thực tế có rất nhiều hệ thống như thế mà cần giao tiếp với REST APIs sử dụng các phương thức HTTP. Trong khi các hệ thống này có giao thức kết nối giống nhau, một trong số đó có thể là một hệ thống mà AIMS Software cần giao tiếp trong tương lai. Vì vậy, vì mục đích tái sử dụng, chúng ta có thể cần một lớp mới, ví dụ là API, để phụ trách giao tiếp API qua phương thức HTTP GET và HTTP POST. Ngoài ra, chúng ta sẽ nhìn nhận các vấn đề ở lớp control trong phần sau.

##### Thiết kế lớp entity

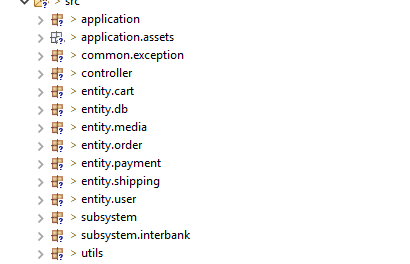
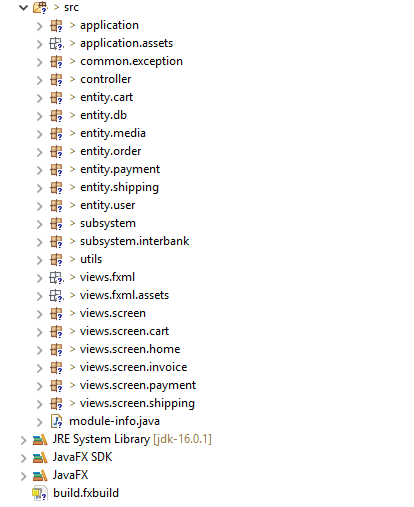
Với bản mô tả hiện tại và đối với 2 use case “Pay Order” và “Place Order”, đa số các lớp entity trong thiết kế kiến trúc đã đơn giản, và có thể ánh xạ 1-1 với lớp thiết kế.

##### Thiết kế lớp control

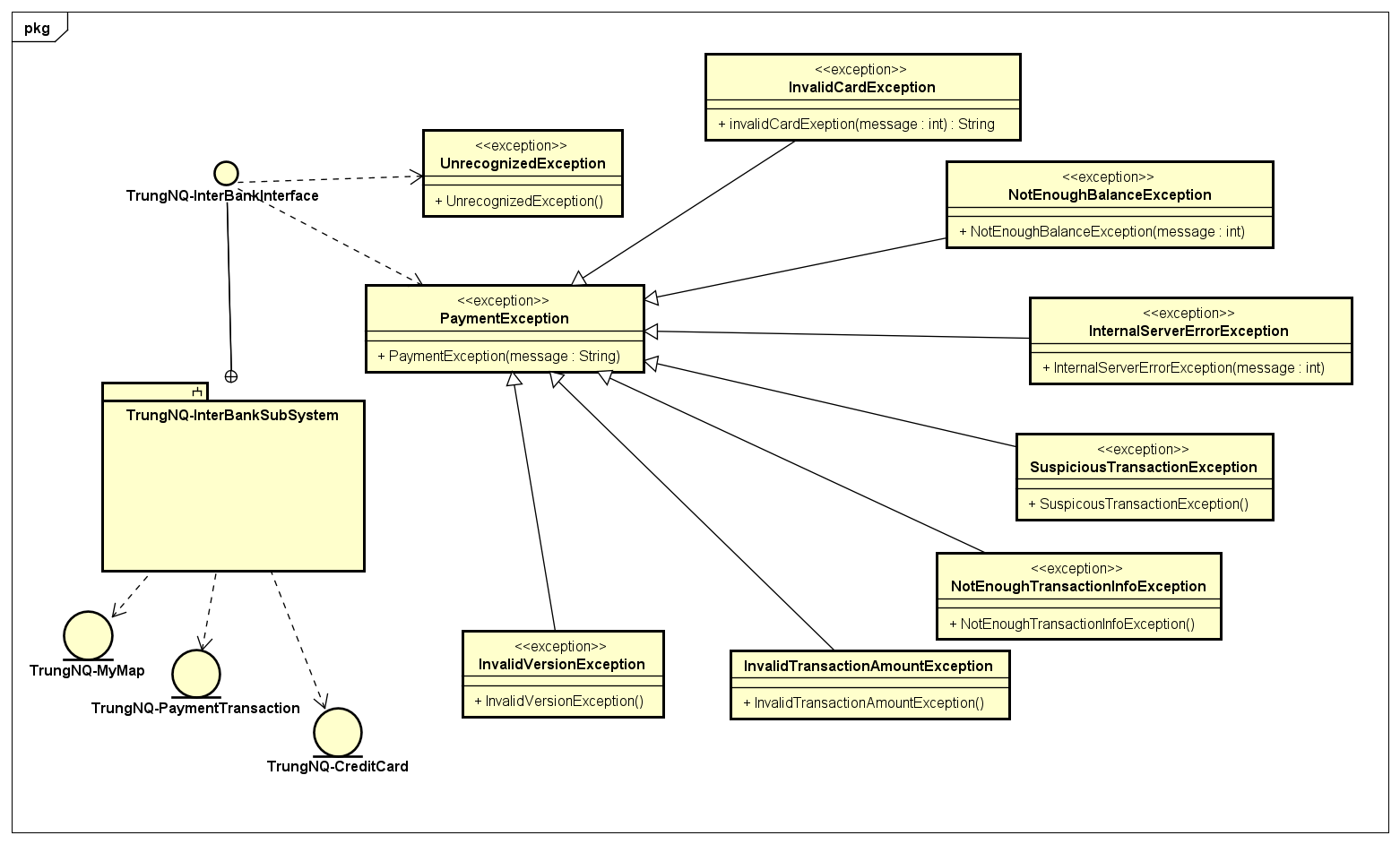
Tương tự, đa số các lớp control trong thiết kế kiến trúc đã đơn giản, và có thể ánh xạ 1-1 với lớp thiết kế. Tuy nhiên, InterbankSubsystemController đang phụ trác 2 tác vụ: (1) điều khiển luồng dữ liệu và (2) chuyển đổi dữ liệu (chuyển đổi dữ liệu sang định dạng yêu cầu và xử lý kết quả trả về). Do đó chúng ta cần ít nhất một lớp khác để phụ trách chuyển đổi dữ liệu, ví dụ JSON hoặc MyMap (tùy thuộc vào thiết kế mà lớp này có thể tái sử dụng cho các hệ thống thông tin web khác).

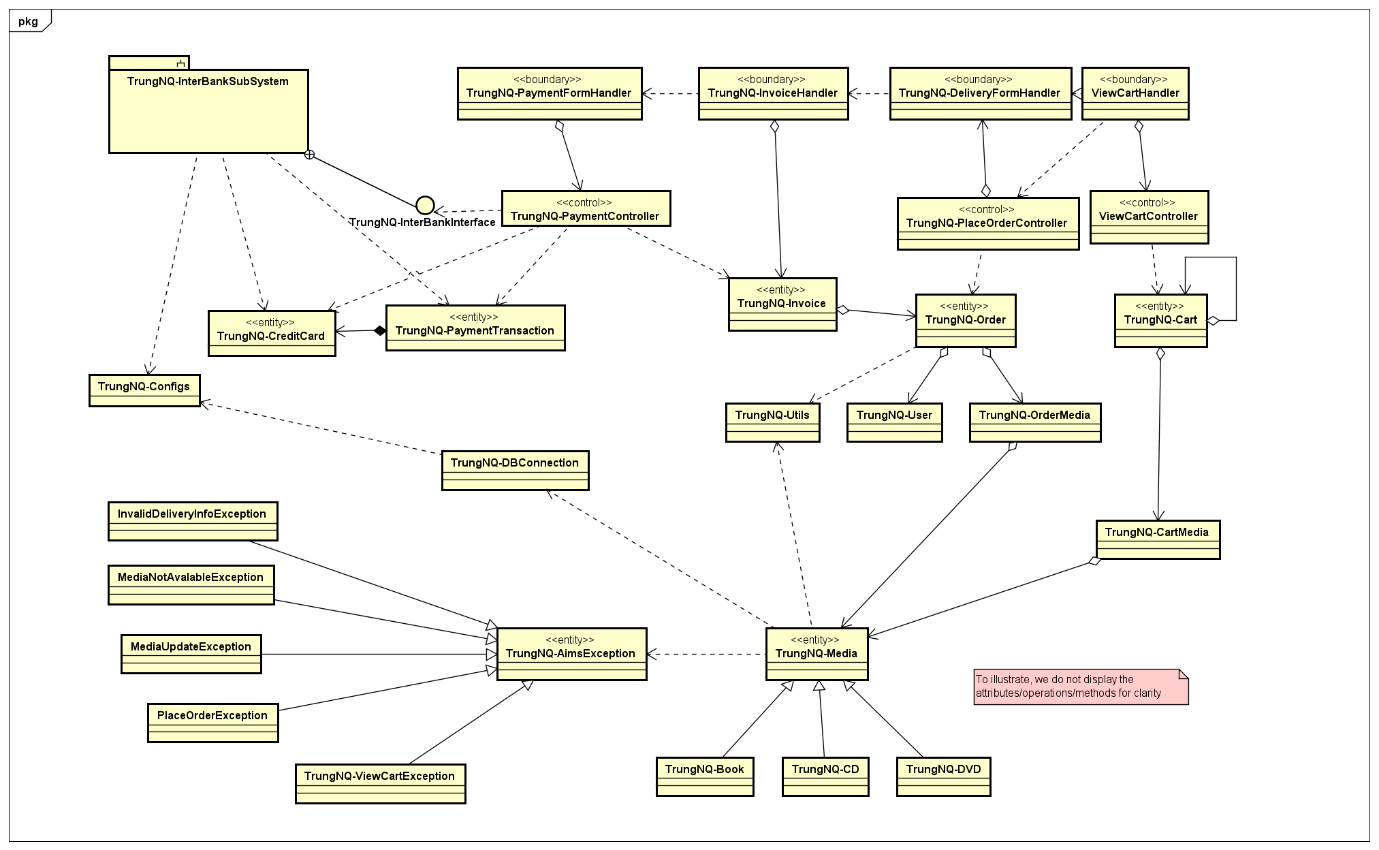
##### Nhóm các lớp thiết kế

Sau đây là một cách để nhóm các lớp thiết kế vào các package.

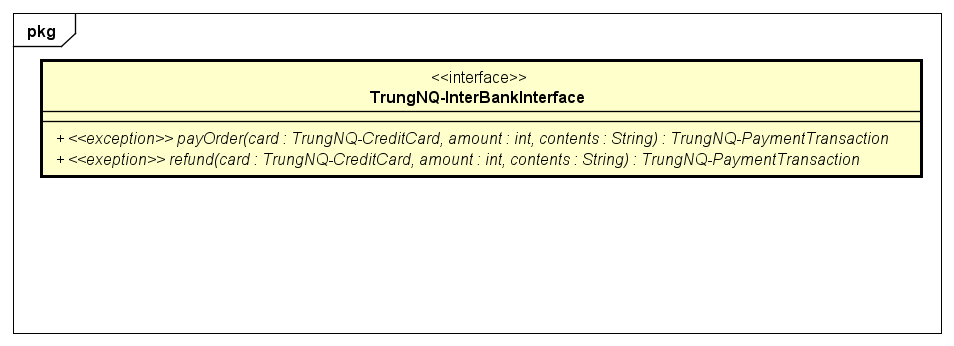


Xác định mối quan hệ giữa các lớp:

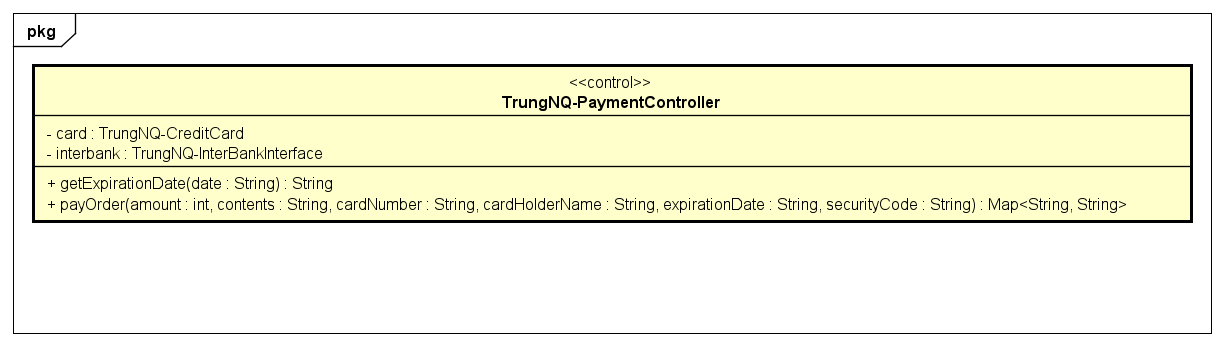




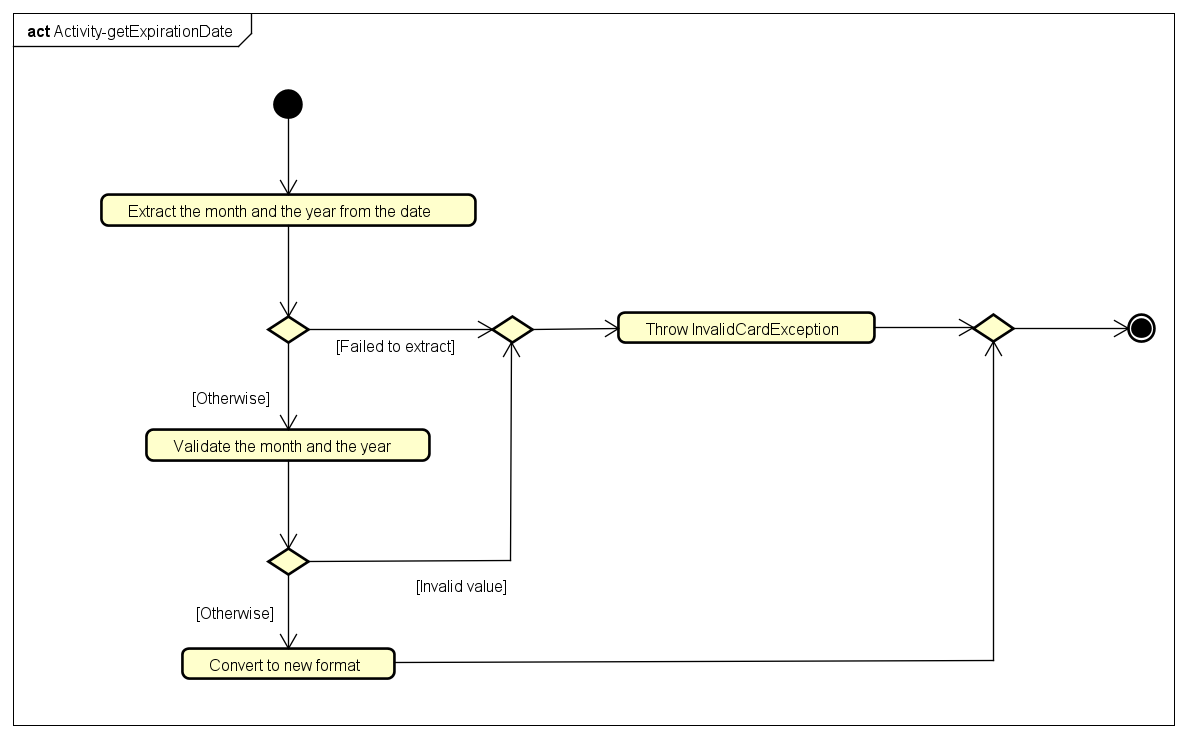
Method TrungNQ-InterbankInterface



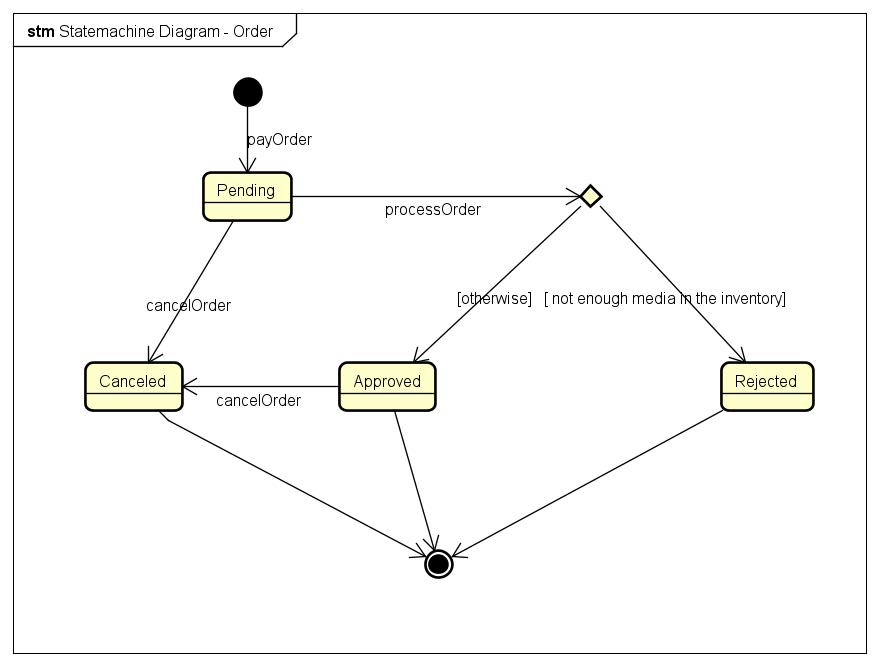
Method TrungNQ-PaymentController



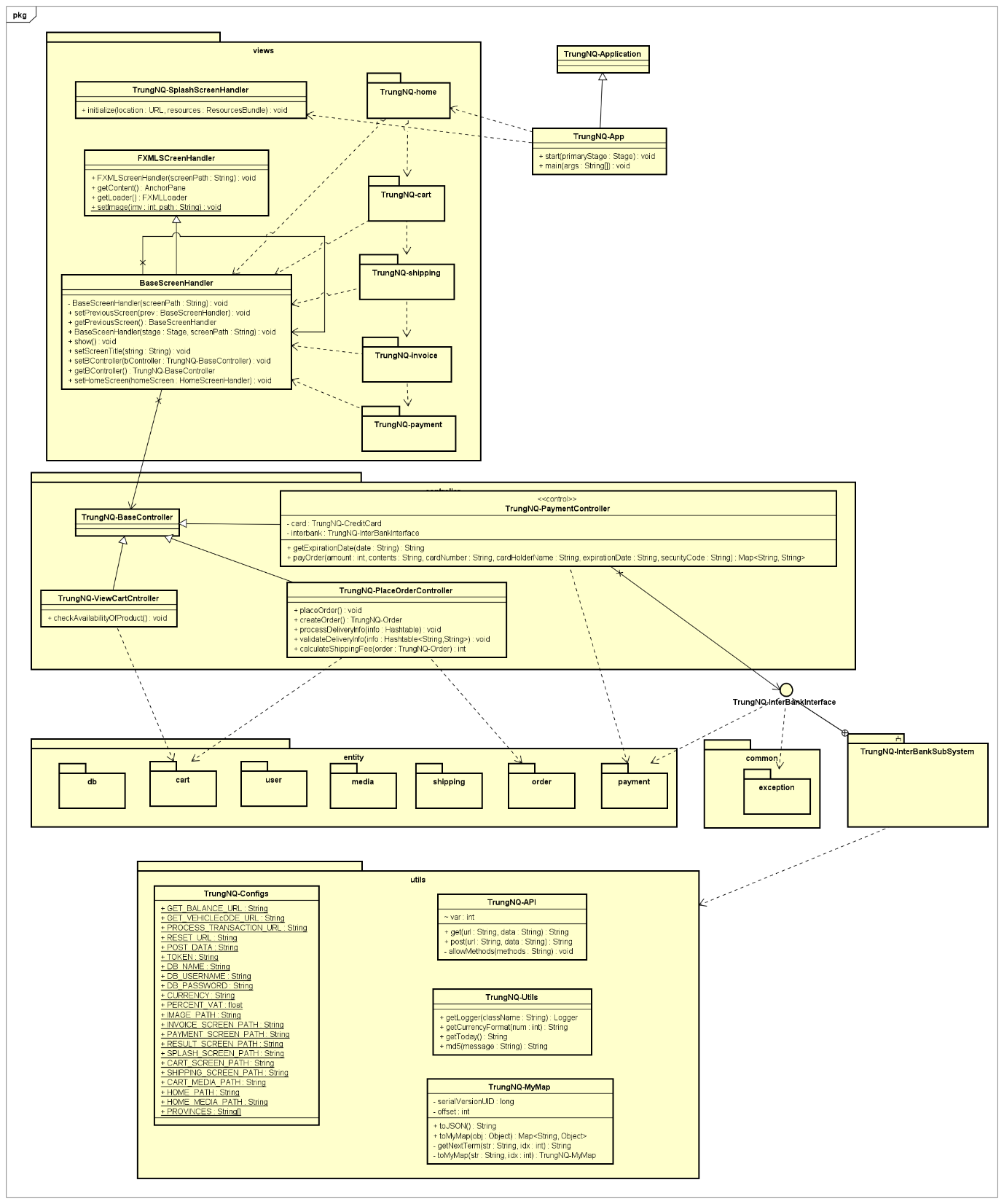
Activity diagram getExpiratinDate

******

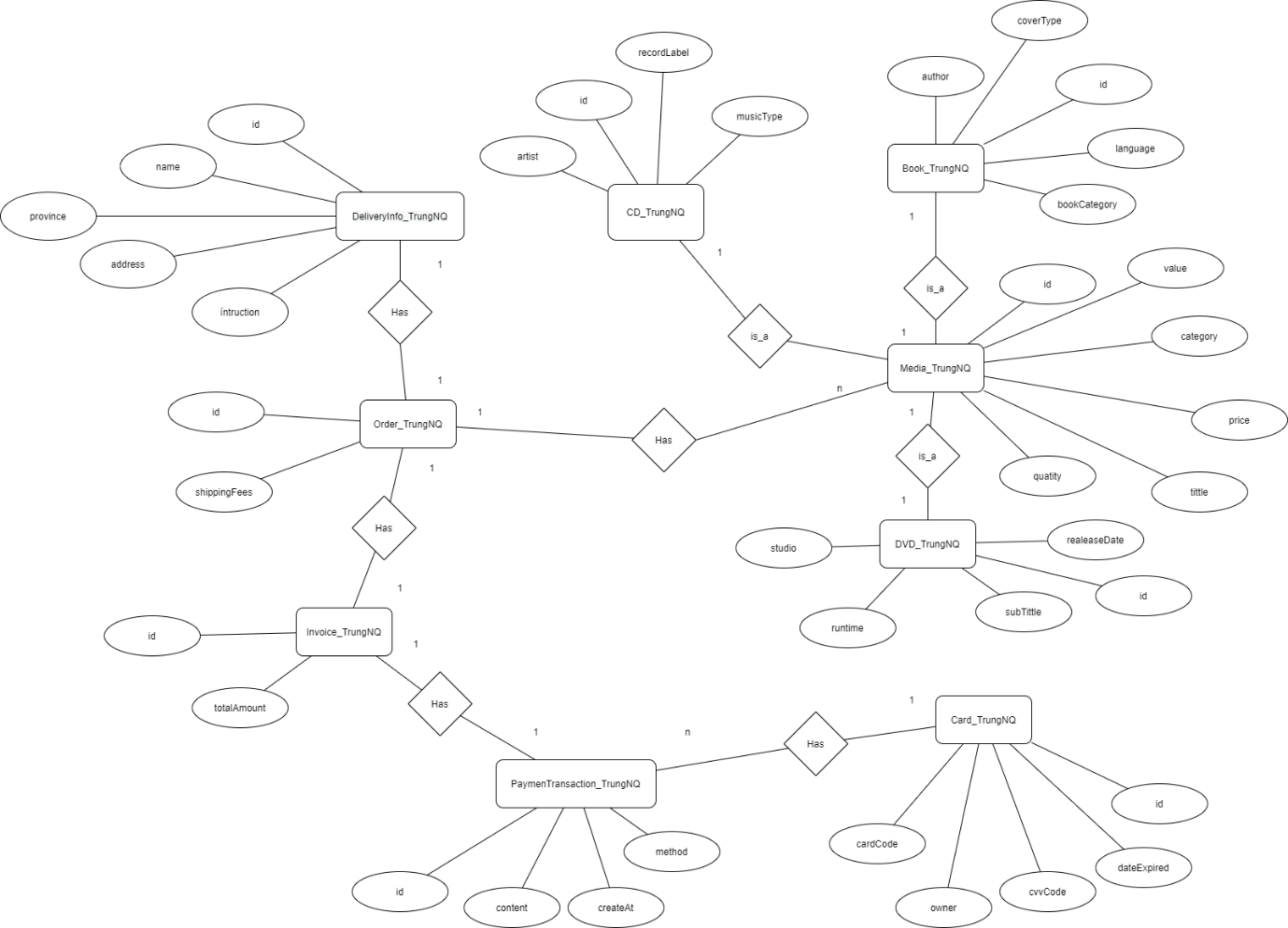
State machine Order



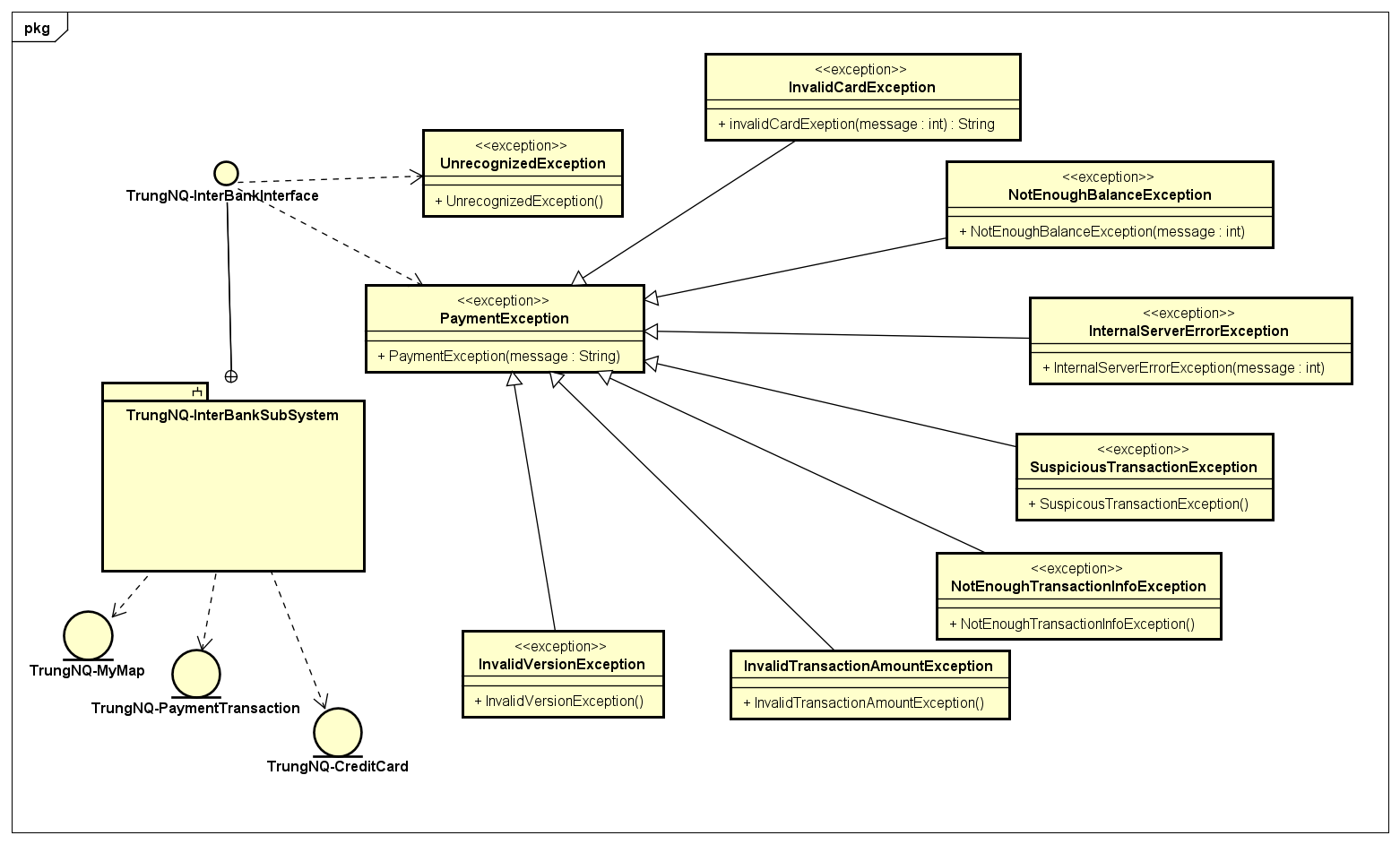
Biểu đồ lớp thiết kế

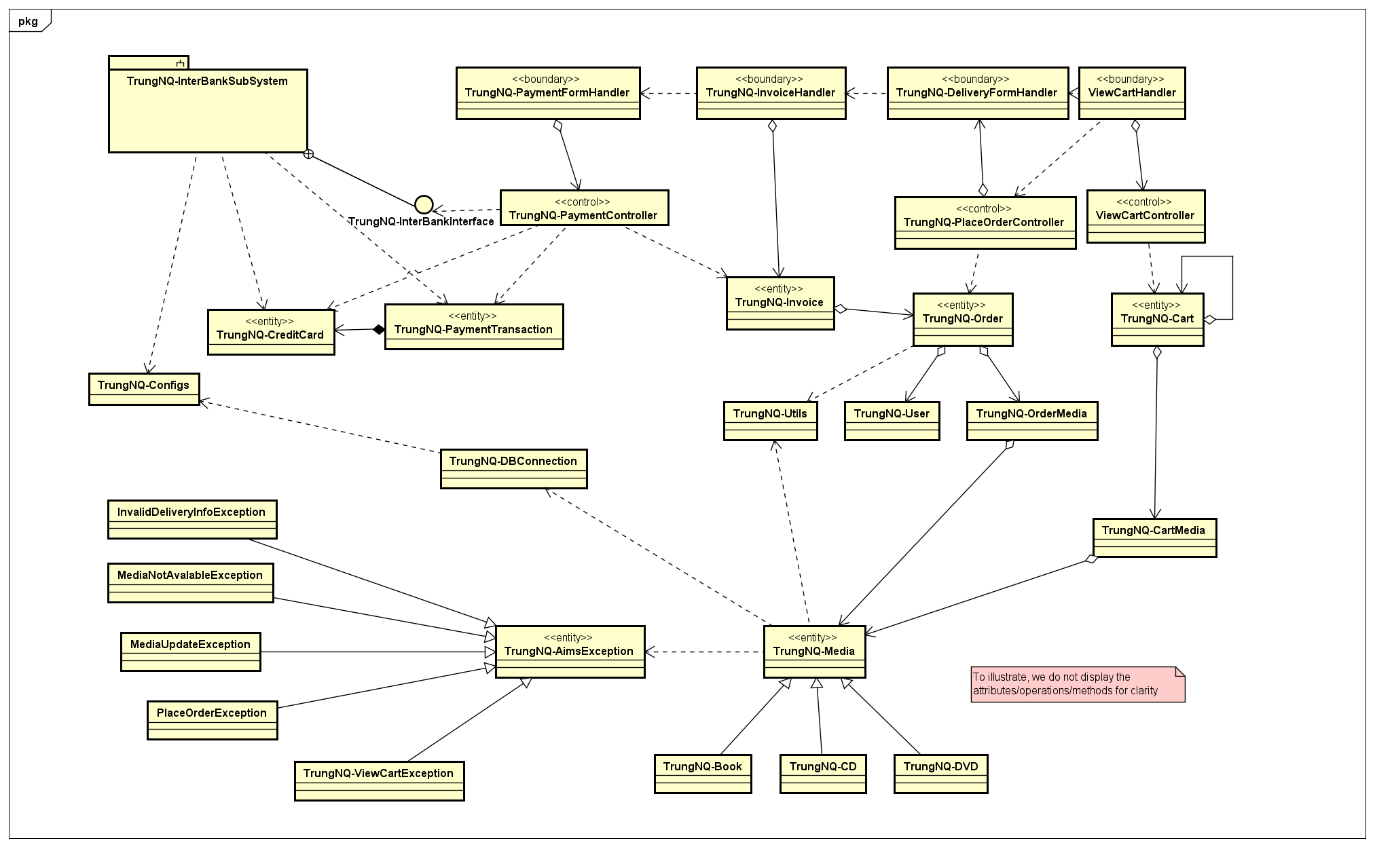


Mô hình hóa khái niệm:

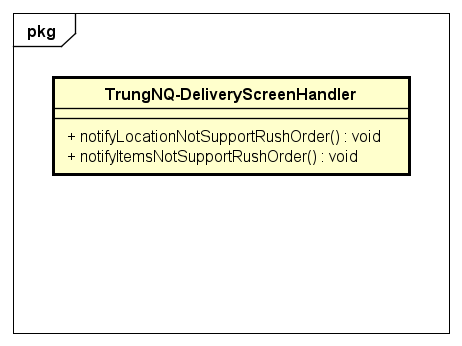


Mối quan hệ các lớp có PlaceRushOrder





Method TrungNQ-DeliveryScreenHandler



Method TrungNQ-PlaceRushOrderController

