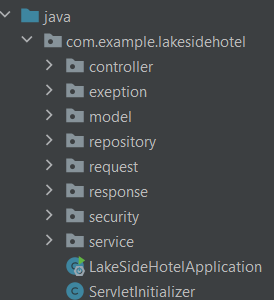
**Tổng quan cấu trúc File bên Back-End:**

- Controller: chịu trách nhiệm xử lý các yêu cầu HTTP từ client, định tuyến các yêu cầu này tới các dịch vụ tương ứng, và trả về kết quả. Nó sẽ nhận yêu cầu từ Clien và ánh xạ vô một phương thức xử lý, gọi các phương thức trong service rồi trả về response.

- Exception: dùng để xử lý các lỗi phát sinh trong phát trình chạy.

- Model: là các lớp để ánh xạ với các bảng trong cơ sử dữ liệu.

- Repository: chịu trách nhiệm giao tiếp với cơ sở dữ liệu, dùng để truy xuất hoặc lưu trữ dữ liệu.

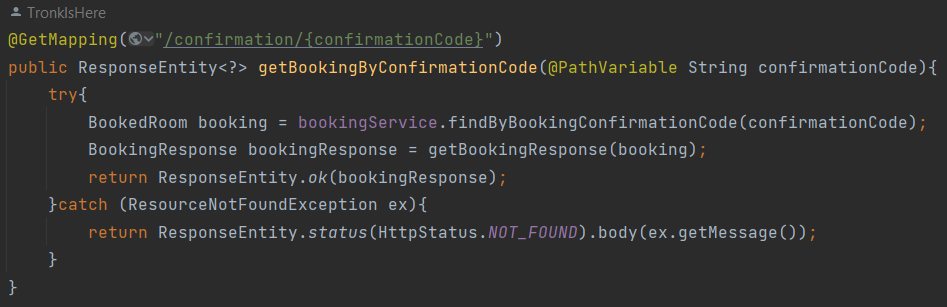
- Request: chứa các lớp biểu diễn dữ liệu của yêu cầu từ client.

- Response: chứa các lớp biểu diễn dữ liệu trả về cho client.

- Security: Chứa các cấu hình và logic bảo mật cho ứng dụng, controller và service tương tác với sercurity để kiểm tra quyền truy cập của người dùng.

- Service: Chứa các file methods cũng như Implements.

**Ví dụ về cách một động của một hàm trong Controller:**

****

Hàm sẽ nhận yêu cầu HTTP với Url là “/confirmation/{confirmationCode}”, với dữ liệu được truyền vào là confermationCode (mã xác nhận để mà tìm phòng dựa trên nó), hàm sẽ tạo một biến tên booking dựa vào lớp BookedRoom trong pagckage model, nhưng biến đó sẽ có dữ liệu dựa trên hàm findByBookingConfimationCode của lớp bookingSerivce trong packageService.

Biến booking có được sẽ có là booking có mã số xác nhận được truyền vào. Cuối cùng dữ liệu được trả lại Client sẽ là dữ liệu được tạo từ BookingResponse với hàm lấy Response (getBookingResponse) dựa trên booking đã lấy được, từ đó trả lại toàn bộ dữ liệu.

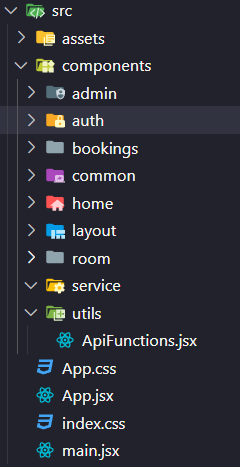
Tại sao không trả thẳng dữ liệu dựa vào lớp Model mà thay vào đó lại phải từ Response:

- Tách biệt giữa tầng dữ liệu và tầng hiển thị: Model sẽ đại diện cho cấu trúc dữ liệu trong cơ sở dữ liệu, trong khi Response đại diện cho dữ liệu mà API trả về cho client. Và Response giúp kiểm soát chính xác những gì được trả về từ đó nâng cao tính bảo mật.

- Linh hoạt trong việc định dạng kiểu dữ liệu trả về: có thể dễ dàng tùy chỉnh dữ liệu trả về trong các lớp Response, các thông tin bổ xung không có trong model.

- Dễ dàng mở rộng và bảo trì (cái này cần học thêm).

- Xử lý lỗi và trạng thái dễ dàng hơn, vì các lớp Respinse có thể bao gồm thông tin trạng thái, thông báo một cách rõ ràng, điều này không thể thực hiện nếu chỉ trả về đối tượng Model.

**** **Tổng quan cấu trúc file Front-end:**

React sẽ hoạt động bằng cách chia giao diện người dùng thành các thành phần có thể tái sử dụng khác nhau thường được xử lý độc lập.

Ví dụ về cách hoạt động components của React trong project: khi đặt phòng thì component Checkout sẽ được gọi, bên trong Checkout sẽ gọi thêm 2 conponent khác là BookingForm và RoomCarousel, khi chúng ta kích hoạt BookingForm thì BookingSuccess sẽ được gọi ra.

Xét về tổng quan sẽ khá rối rắm nên một roadmap sẽ được soạn ra và cập nhật trong thời gian sắp tới.

**Dự định hiện tại:**

Nâng cấp giao diện:

- Add-room

- Edit-room

- Existing-rooms

- book-room

- browse-all-rooms

- admin

- booking-success

- existing-booking

- find-booking

- login

- register

- profile

- logout

=> Mục tiêu chính bao gồm: tìm được thiết kế phù hợp với web đặt khách sạn, cải tiến giao diện phù hợp hơn với người dùng

Nâng cấp chức năng:

- Nâng cấp Database để mỗi phòng được đặt thuộc về một người dùng nhất định (dựa vào Id người dùng).

- Cho người dùng khả năng chỉnh sửa thông tin cá nhân của bản thân cũng như là để Avt.

=> Gia tăng trải nghiệm người dùng.

**Cập nhật tình hình hiện tại (24/5/2024):**

**Dự định hiện tại:**

Nâng cấp giao diện:

- Existing-rooms

- book-room

- admin

- booking-success

- profile

=> Mục tiêu chính bao gồm: cải tiến để hợp với tiến độ đồ án

Nâng cấp chức năng:

- Nâng cấp Database để mỗi phòng được đặt thuộc về một người dùng nhất định (dựa vào Id người dùng), mỗi phòng sẽ có miêu tả phòng.

- Cho người dùng khả năng chỉnh sửa thông tin cá nhân của bản thân cũng như là để Avt.

=> Gia tăng trải nghiệm người dùng.

**Đồ án tốt nghiệp:**

- Viết báo cáo.