

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG  
KHOA CÔNG NGHỆ THÔNG TIN



Bài Tập Lớn

**Kiến trúc và thiết kế hệ thống phần mềm**

**ĐỀ TÀI**

**Hệ thống quản lý rạp chiếu phim**

Nhóm học phần: 04

Nhóm BTL: 02

**Sinh viên: Nguyễn Đức Anh – B20DCCN056**

**Các Module đăng ký:** Quản lý khách hàng, Chức năng khách  
hàng mua vé online, Thống kê khách hàng theo doanh thu

<b>1. Hoạt động của các module</b>	<b>2</b>
1.1. Quản lý thông tin khách hàng (Xem thông tin khách hàng, sửa thông tin khách hàng)	2
1.2 Khách hàng mua vé online	3
1.3 Quản lý xem thống kê khách hàng theo doanh thu	4
<b>2. Thiết kế thực thể (module chung)</b>	<b>5</b>
<b>3. Thiết kế cơ sở dữ liệu (module chung)</b>	<b>7</b>
<b>4. Module Quản lý thông tin khách hàng</b>	<b>8</b>
4.1 Thiết kế giao diện	8
4.2 Thiết kế biểu đồ lớp chi tiết	10
4.3 Thiết kế biểu đồ tuần tự	11
<b>5. Module Khách hàng mua vé online</b>	<b>14</b>
5.1 Thiết kế giao diện	14
5.2 Thiết kế biểu đồ lớp chi tiết	18
5.3 Thiết kế biểu đồ tuần tự	19
<b>6. Module Khách hàng mua vé online</b>	<b>23</b>
6.1 Thiết kế giao diện	23
6.2 Thiết kế biểu đồ lớp chi tiết	24
6.3 Thiết kế biểu đồ tuần tự	26

## 1. Hoạt động của các module

1.1. Quản lý thông tin khách hàng (Xem thông tin khách hàng, sửa thông tin khách hàng)

- Quản lý truy cập vào hệ thống, hệ thống hiện lên giao diện đăng nhập
  - Ô nhập Số điện thoại
  - Ô nhập mật khẩu
  - Nút đăng nhập
- Quản lý nhập đúng thông tin số điện thoại và mật khẩu và click nút đăng nhập -> giao diện menu quản lý hiện lên:
  - Nút quản lý thông tin khách hàng
  - Nút xem thống kê khách hàng
  - ...

- Quản lý click nút quản lý thông tin khách hàng -> giao diện tùy chọn hiện ra
  - Nút thêm khách hàng mới
  - Nút tìm kiếm khách hàng
- Quản lý click nút tìm kiếm khách hàng -> giao diện tùy chọn hiện ra
  - Ô tìm kiếm tên khách hàng, mã khách hàng
  - Nút tìm kiếm
- Quản lý nhập tên khách hàng đã tồn tại và bấm tìm kiếm -> giao diện hiện ra danh sách khách hàng có tên giống với tên quản lý tìm kiếm.
  - Mỗi khách hàng là một button, có thông tin: sđt, tên
- Quản lý click vào một khách hàng -> giao diện hiện ra thông tin chi tiết của khách hàng:
  - Số điện thoại (chỉ xem)
  - Tên(xem và sửa)
  - Ngày sinh(xem và sửa)
  - Địa chỉ(xem và sửa)
  - Nút khóa tài khoản
  - Nút cập nhật tài khoản khách hàng
- Quản lý sửa các thông tin của khách hàng theo đúng yêu cầu và bấm nút cập nhật tài khoản -> giao diện hiển thị lại thông tin của khách hàng khi cập nhật thành công

## 1.2 Khách hàng mua vé online

- Khách hàng truy cập vào hệ thống, hệ thống hiện lên giao diện đăng nhập
  - Ô nhập số điện thoại
  - Ô nhập mật khẩu
  - Nút đăng nhập
- Khách hàng nhập đúng thông tin số điện thoại và mật khẩu và click nút đăng nhập -> giao diện menu khách hàng hiện lên:
  - Nút mua vé phim online
  - Nút xem thông tin cá nhân
  - Nút xem lịch sử đặt mua
- Khách hàng click nút mua vé phim online -> hệ thống hiện lên giao diện danh sách các phim đang mở bán vé

- Mỗi phim là một button, hiển thị các thông tin tên phim, thời lượng, thể loại, nội dung ..
- Khách hàng click vào một bộ phim -> hệ thống hiện lên giao diện chi tiết bộ phim khách hàng vừa click
  - Tên phim, thời lượng, thể loại, ...
  - Danh sách các lịch chiếu của bộ phim đấy (mỗi suất chiếu là một nút, hiển thị ngày chiếu, thời gian bắt đầu, thời gian kết thúc, còn vé hay hết vé)
  - Nút quay lại
- Khách hàng click vào một lịch chiếu cụ thể -> hệ thống hiện lên giao diện thông tin của lịch chiếu, ngày chiếu, thời gian chiếu, thời gian bắt đầu chiếu, thời gian kết thúc và các vé phim của lịch chiếu đấy, mỗi một vé là một nút với các thông tin
  - ID vé
  - Vị trí ghế
  - Giá tiền
  - Nút đặt vé
    - Và hiển thị thông tin phòng chiếu của lịch chiếu đấy
  - Tên phòng chiếu
- Khách hàng click chọn tick một hoặc nhiều vé -> hệ thống hiển thị giá tiền
- Khách hàng click nút đặt vé sau khi chọn tick -> hệ thống hiển thị giao diện đặt vé online thành công:
  - Nút đặt mua tiếp
  - Nút quay về trang chủ

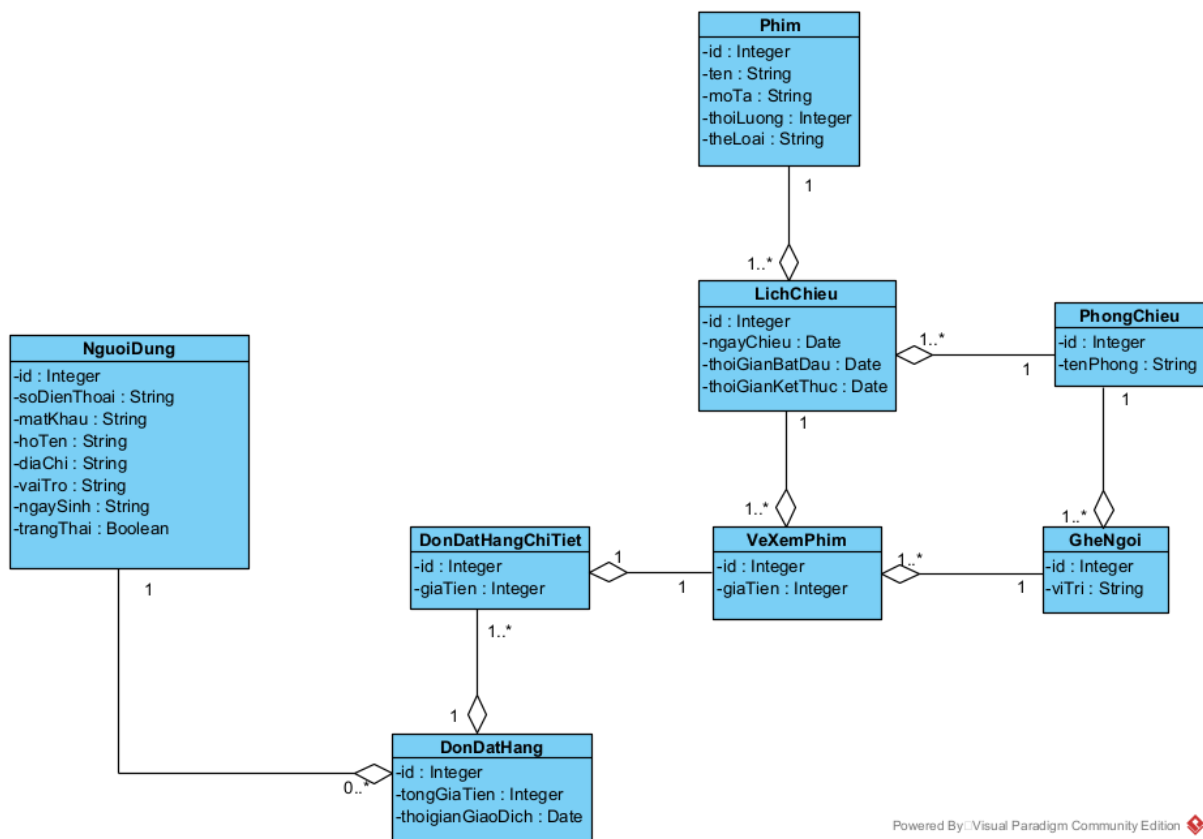
### 1.3 Quản lý xem thống kê khách hàng theo doanh thu

- Quản lý truy cập vào hệ thống, hệ thống hiện lên giao diện đăng nhập
  - Ô nhập Số điện thoại
  - Ô nhập mật khẩu
  - Nút đăng nhập
- Quản lý nhập đúng thông tin số điện thoại và mật khẩu và click nút đăng nhập -> giao diện menu quản lý hiện lên:
  - Nút quản lý thông tin khách hàng
  - Nút xem thống kê khách hàng
  - Nút xem thông tin cá nhân

- Quản lý click nút xem thống kê khách hàng -> giao diện hiện lên danh sách các khách hàng và số tiền và khách hàng đó đã đặt mua vé online
  - Mỗi phần tử trong danh sách là một button
- Quản lý click chi tiết vào button hiển thị tên khách hàng trên -> giao diện hiện ra thông tin cá nhân của khách hàng đấy(chỉ hiển thị, không thể chỉnh sửa) và danh sách các đơn mua vé của khách hàng đấy:
  - Tổng giá tiền
  - Thời gian giao dịch
  - Mỗi đơn mua vé lại có danh sách các vé đã mua chi tiết (ở module 1.2) và hiển thị chi tiết các vé mua chi tiết, mỗi vé mua chi tiết gồm gồm giá tiền, ghế ngồi

## 2. Thiết kế thực thể (module chung)

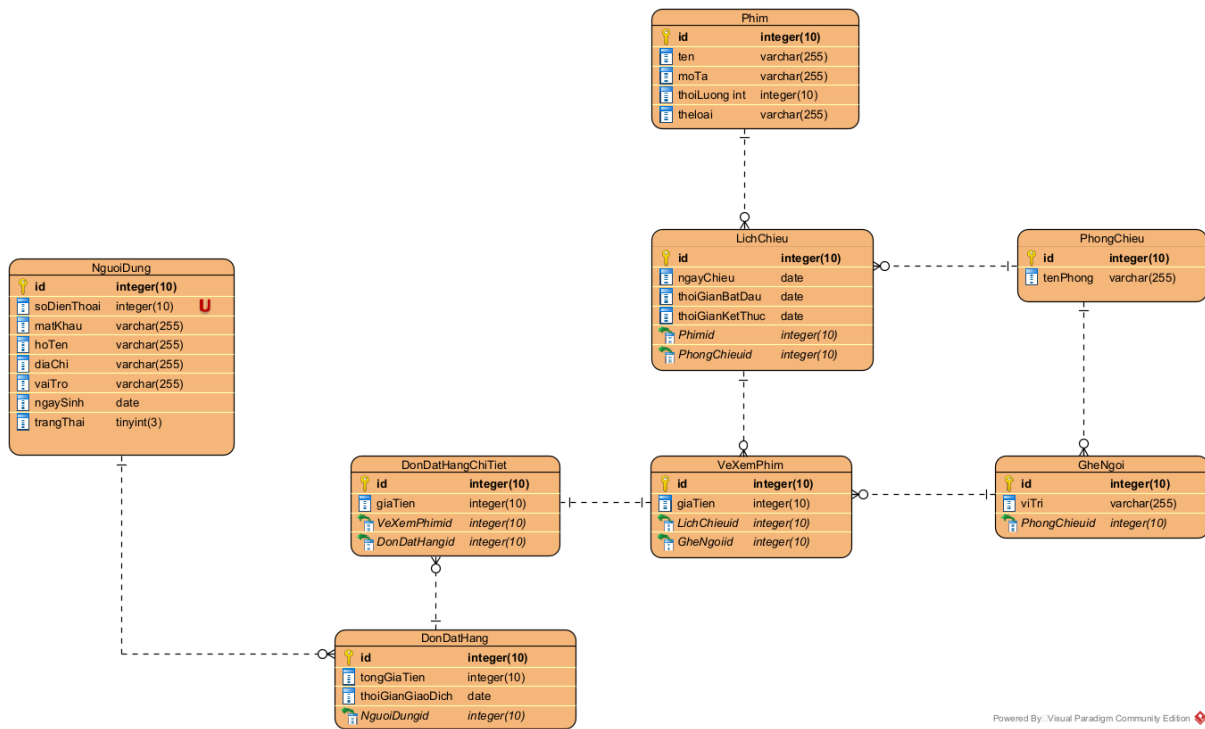
- Trong module **Quản lý thông tin khách hàng** có các thực thể:
  - **NguoIDung**: id, soDienThoai, matKhau, hoTen, diaChi, vaiTro, ngaySinh, trangThai(có hoạt động hay bị khóa) => để lưu thông tin khách hàng và quản lý
- Trong module **Khách hàng mua vé online** có các thực thể:
  - **Phim**: id, ten, moTa, thoiLuong, theLoai => để lưu thông tin Phim
  - **LichChieu**: id, ngayChieu, thoiGianBatDau, thoiGianKetThuc => để lưu thông tin lịch chiếu phim
  - **PhongChieu**: id, tenPhong => để lưu thông tin phòng chiếu
  - **VeXemPhim**: id, giaTien => để lưu thông tin vé xem phim của lịch chiếu
  - Bổ sung thêm **GheNgoi**: id, viTri => để lưu thông tin ghế ngồi
  - Với mỗi khách hàng có thể có nhiều lần mua vé khác nhau, => cần tạo 1 thực thể **DonDatHang** với quan hệ *NguoIDung 1 - N DonDatHang*(id, tongGiaTien, thoigianGiaoDich) để lưu lại lịch sử đặt hàng
  - Với mỗi một lần đặt hàng có thể đặt nhiều vé => Tạo 1 thực thể Đơn đặt hàng chi tiết **DonDatHangChiTiet**(id, giaTien) để lưu chi tiết 1 vé xem phim được mua(trong nhiều vé của 1 lần đặt hàng mua vé)



Powered By: Visual Paradigm Community Edition

- Một bộ phim có thể có nhiều lịch chiếu khác nhau, quan hệ 1 Phim - N LichChieu
- Một phòng chiếu có nhiều lịch chiếu ở các thời điểm các nhau, quan hệ 1 PhongChieu - N LichChieu
- Một phòng chiếu có nhiều ghế ngồi, quan hệ 1 PhongChieu - N GheNgoi
- Một lịch chiếu có nhiều Vé xem phim, quan hệ 1 LichChieu - N VeXemPhim
- Một Ghế ngồi có nhiều Vé xem phim(do có nhiều lịch chiếu của 1 phòng chiếu, nên cũng có nhiều vé xem phim của một phòng chiếu nhưng khác lịch chiếu) quan hệ 1 GheNgoi - N VeXemPhim
- 1 DonDatHangChiTiet - 1 VeXemPhim
- Một Đơn đặt hàng sẽ có nhiều Đơn đặt hàng chi tiết(do một thời điểm đặt hàng có thể chọn mua nhiều vé xem phim), quan hệ 1 DonDatHang - N DonDatHangChiTiet
- Một Người dùng có thể có nhiều lần mua hàng, 1 NguiDung - N DonDatHang

### 3. Thiết kế cơ sở dữ liệu (module chung)



- Diễn giải:
- Mỗi lớp thực thể ở trên đề xuất tạo thành 1 bảng trong cơ sở dữ liệu tương ứng và kiểu dữ liệu của mỗi bảng phải tương ứng với kiểu dữ liệu của các thuộc tính của lớp thực thể:
  - Lớp Phim -> bảng Phim
  - Lớp LichChieu -> bảng LichChieu
  - Lớp PhongChieu -> bảng PhongChieu
  - Lớp GheNgoi -> bảng GheNgoi
  - Lớp VeXemPhim -> bảng VeXemPhim
  - Lớp DonDatHangChiTiet -> bảng DonDatHangChiTiet
  - Lớp DonDatHang -> bảng DonDatHang
  - Lớp NguiDung -> bảng NguiDung(cần lưu ý soDienThoai phải là unique)
- Chuyển quan hệ số lượng giữa các thực thể thành quan hệ số lượng giữa các bảng và bổ sung thuộc tính khóa(khóa chính được thiết lập với thuộc tính id của các bảng), khóa ngoại được thiết lập cho các bảng:
  - 1 Phim - N LichChieu => bảng LichChieu có khóa ngoại Phimid
  - 1 PhongChieu - N LichChieu => bảng LichChieu có khóa ngoại PhongChieuid

- 1 PhongChieu - N GheNgoi => bảng GheNgoi có khóa ngoại PhongChieuid
- 1 LichChieu - N VeXemPhim => bảng VeXemPhim có khóa ngoại LichChieuid
- 1 GheNgoi - N VeXemPhim => bảng VeXemPhim có khóa ngoại GheNgoiid
- 1 DonDatHangChiTiet - 1 VeXemPhim (nhưng theo nghiệp vụ và quan hệ thực thể đã vẽ, DonDatHangChiTiet sẽ chứa VeXemPhim) => bảng DonDatHangChiTiet có khóa ngoại VeXemPhimid
- 1 DonDatHang - N DonDatHangChiTiet => bảng DonDatHangChiTiet có khóa ngoại DonDatHangid
- 1 NguoiDung - N DonDatHang => bảng DonDatHang có khóa ngoại NguoiDungid

## 4. Module Quản lý thông tin khách hàng

### 4.1 Thiết kế giao diện

#### - **Giao diện đăng nhập:**

Login

Số điện thoại

Mật khẩu

Đăng nhập

#### - **Giao diện menu quản lý:**

Hệ thống quản lý phim, MANAGER Nguyễn Đức Anh!

Quản lý thông tin khách hàng

Xem Thông Kê Khách Hàng

Xem Thông Tin Cá Nhân

#### - **Giao diện quản lý thông tin khách hàng:**



# Quản Lý Thông Tin Khách Hàng

- Thêm Khách Hàng Mới
- Tìm Kiếm Khách Hàng

- Giao diện tìm kiếm khách hàng:

Tìm Kiếm Khách Hàng

Tên Khách Hàng

Tìm Kiếm

Tìm Kiếm Khách Hàng

Tên Khách Hàng

Tìm Kiếm

Kết Quả Tìm Kiếm		
Tên	Số điện thoại	Hành động
Nguyễn Đức Anh	0987838802	Chỉnh sửa
Nguyễn Văn Anh 11	0965777888	Chỉnh sửa

- **Giao diện thông tin khách hàng:**

Số điện thoại

0987838802

Tên

Nguyễn Đức Anh

Ngày sinh

23/05/2024

Địa chỉ

Hà Nội

Cập nhật tài khoản khách hàng

Khóa tài khoản

[illegible]

- Diễn giải: Sử dụng mô hình MVC, chia thành các tầng view, model, controller và xử lý lần lượt từng tầng theo thứ tự từ trên xuống dưới
  - Cần xây dựng các giao diện HTML tương ứng với phần trên: login.html, menuQuanLy.html, quanLyKhachHang.html, thôngTinKhachHang.html
  - Tạo lớp **AuthController** để nhận các request đăng nhập từ người dùng, lớp này inject AuthService, với các method getDangNhap(), postDangNhap nhận vào soDienThoai, matKau

- Tạo lớp **ManagerController** để nhận các request từ người dùng khi chọn giao diện menu quản lý, menu quản lý khách hàng với các method `getPageQuanLy()`, `getPageQuanLyKhachHang()`
- Tạo lớp **UserController** để nhận các request từ người dùng khi muốn lấy danh sách các User, xem chi tiết 1 User, cập nhật chi tiết 1 User, lớp này inject `UserController`, với các method `findUser()`, `detailUser()`, `updateUser()`
- Tạo lớp **AuthService** có inject `UserRepository` (kế thừa `JpaRepository`) để xử lý các yêu cầu đăng nhập từ tầng `AuthController`, với method `dangNhap()` nhận 2 tham số `username`, `password` và trả về đăng nhập thành công hay không thành công
- Tạo lớp **UserService** có inject `UserRepository` (kế thừa `JpaRepository`) để xử lý các yêu cầu lấy danh sách người dùng *method* `findUser` (có thể filter theo name), xem chi tiết 1 người dùng cụ thể *method* `DetailUser()` (nhận vào id), cập nhật chi tiết 1 người dùng *method* `updateUser()` (nhận vào id, và các field cần cập nhật)
- Tạo interface **UserRepository** kế thừa `JpaRepository`, dùng để thực hiện các câu truy vấn liên quan tới lớp **NguoDung**

#### 4.3 Thiết kế biểu đồ tuần tự



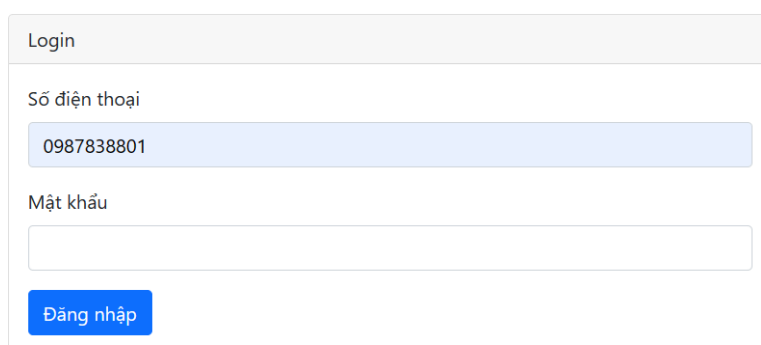
16. Lớp AuthController trả về menuQuanLy.html
17. menuQuanLy.html hiện lên giao diện menu của quản lý
18. quản lý click vào nút quản lý khách hàng
19. menuQuanLy.html gửi yêu cầu tới ManagerController, method = GET, endpoint = /quanLyKhachHang
20. Lớp ManagerController thực hiện hàm getPageQuanLyKhachHang()
21. Hàm getPageQuanLyKhachHang() trả về quanLyKhachHang.html
22. quanLyKhachHang.html hiện lên giao diện quản lý khách hàng
23. Quản lý click nút tìm kiếm khách hàng
24. quanLyKhachHang.html gửi request tới ManagerController, method = GET, endpoint = /timKiemKhachHang
25. Lớp ManagerController thực hiện hàm getPageTimKiemKhachHang()
26. Hàm getPageTimKiemKhachHang() trả về timKiemKhachHang.html
27. timKiemKhachHang.html hiện lên giao diện tìm kiếm khách hàng
28. Quản lý nhập tên khách hàng và click nút tìm kiếm
29. timKiemKhachHang.html gửi request tới UserController, method = GET, endpoint = /user/all?name=
30. Lớp UserController thực hiện hàm findUser()
31. Hàm findUser() của lớp UserController gọi tới lớp UserService
32. Lớp UserService thực hiện hàm findUser()
33. Hàm findUser() của lớp UserService gọi tới UserRepository
34. UserRepository gọi tới lớp NguoiDung
35. Lớp NguoiDung tự đóng gói đối tượng
36. Lớp NguoiDung trả về đối tượng cho UserRepository
37. UserRepository trả về đối tượng cho lớp UserService
38. Lớp UserService trả về đối tượng cho lớp UserController
39. Lớp UserController trả về timKiemKhachHang.html
40. timKiemKhachHang.html hiện lên giao diện tìm kiếm cùng với danh sách các khách hàng đã tìm kiếm
41. Quản lý click vào một khách hàng cụ thể
42. timKiemKhachHang.html gửi request tới UserController, method = GET, endpoint = /user/id
43. Lớp UserController thực hiện hàm detailUser()
44. Hàm detailUser() của lớp UserController gọi tới lớp UserService
45. Lớp UserService thực hiện hàm detailUser()
46. Hàm detailUser() của lớp UserService gọi tới UserRepository
47. UserRepository gọi tới lớp NguoiDung

- 48. Lớp `NguoiDung` tự đóng gói đối tượng
- 49. Lớp `NguoiDung` trả về đối tượng cho `UserRepository`
- 50. `UserRepository` trả về đối tượng cho `UserService`
- 51. Lớp `UserService` trả về đối tượng cho `UserController`
- 52. Lớp `UserController` trả về `thongTinKhachHang.html`
- 53. `thongTinKhachHang.html` hiện lên giao diện thông tin chi tiết của 1 khách hàng
- 54. Quản lý sửa thông tin và click nút cập nhật tài khoản khách hàng
- 55. `thongTinKhachHang.html` gửi request tới `UserController`, `method = POST`, `endpoint = /user/id`
- 56. Lớp `UserController` thực hiện hàm `updateUser()`
- 57. Hàm `updateUser()` của lớp `UserController` gọi tới lớp `UserService`
- 58. Lớp `UserService` thực hiện hàm `updateUser()`
- 59. Hàm `updateUser()` của lớp `UserService` gọi tới `UserRepository`
- 60. `UserRepository` tự thực hiện hàm `save()` và gọi tới `NguoiDung`
- 61. Lớp `NguoiDung` tự đóng gói đối tượng
- 62. Lớp `NguoiDung` trả về đối tượng cho `UserRepository`
- 63. `UserRepository` trả về đối tượng cho `UserService`
- 64. Lớp `UserService` trả về đối tượng cho lớp `UserController`
- 65. Lớp `UserController` trả về `thongTinKhachHang.html`
- 66. `thongTinKhachHang.html` hiện ra thông tin chi tiết khách hàng vừa cập nhật

## 5. Module Khách hàng mua vé online

### 5.1 Thiết kế giao diện

#### - ***Giao diện đăng nhập***



The image shows a login form with a light gray header containing the word "Login". Below the header, there are two input fields. The first is labeled "Số điện thoại" (Phone number) and contains the text "0987838801". The second is labeled "Mật khẩu" (Password) and is empty. At the bottom of the form is a blue button with the text "Đăng nhập" (Login).

#### - ***Giao diện menu khách hàng***

Xin chào, Customer Nguyễn Đức Anh(Customer)!

- [Đặt mua vé online](#)
- [Xem lịch sử đặt mua vé](#)
- [Xem Thông Tin Cá Nhân](#)

- **Giao diện danh sách phim đang mở bán**

Chọn phim để mua vé

**Hành Trình Anh Hùng**  
Thời lượng: 120  
Thể loại: Phiêu lưu, Kỳ ảo  
Một bộ phim phiêu lưu kỳ thú về hành trình của một anh hùng trẻ tuổi.  
[Mua vé](#)

**Tình Yêu Không Biên Giới**  
Thời lượng: 90  
Thể loại: Lãng mạn  
Câu chuyện tình yêu lãng mạn giữa hai người trẻ từ hai thế giới khác nhau.  
[Mua vé](#)

**Cuộc Chiến Bất Tận**  
Thời lượng: 110  
Thể loại: Hành động  
Một bộ phim hành động gay cấn với nhiều pha hành động nghẹt thở.  
[Mua vé](#)

**Hành Trình Của Những Chú Chim**  
Thời lượng: 80  
Thể loại: Tài liệu, Thiên nhiên  
Phim tài liệu về đời sống của loài chim di cư qua các châu lục.  
[Mua vé](#)

- **Giao diện thông tin chi tiết phim**

Hành Trình Anh Hùng

Thời lượng: 120

Thể loại: Phiêu lưu, Kỳ ảo

Nội dung: Một bộ phim phiêu lưu kỳ thú về hành trình của một anh hùng trẻ tuổi.

Lịch Chiếu:

2024-04-16

Bắt đầu: 18:00

Kết thúc: 20:00

Đặt vé

2024-04-16

Bắt đầu: 10:00

Kết thúc: 12:00

Đặt vé

2024-04-16

Bắt đầu: 15:00

Kết thúc: 17:00

Đặt vé

2024-04-17

Bắt đầu: 14:00

Kết thúc: 16:00

Đặt vé

Quay lại

- ***Giao diện thông tin chi tiết của lịch chiếu***



Phòng Chiếu 1

Phòng chiếu: Phòng Chiếu 1

Ngày chiếu: 2024-04-16

Thời gian chiếu: 18:00 - 20:00

Danh sách vé phim

☒

ID vé: 1

Vị trí ghế: A10

Giá tiền: 50000

☐

ID vé: 2

Vị trí ghế: A11

Giá tiền: 50000

☒

ID vé: 3

Vị trí ghế: A12

Giá tiền: 50000

☐

ID vé: 4

Vị trí ghế: B10

Giá tiền: 70000

Tổng giá tiền: 100000

Thanh toán

Quay lại

- **Giao diện thanh toán thành công**

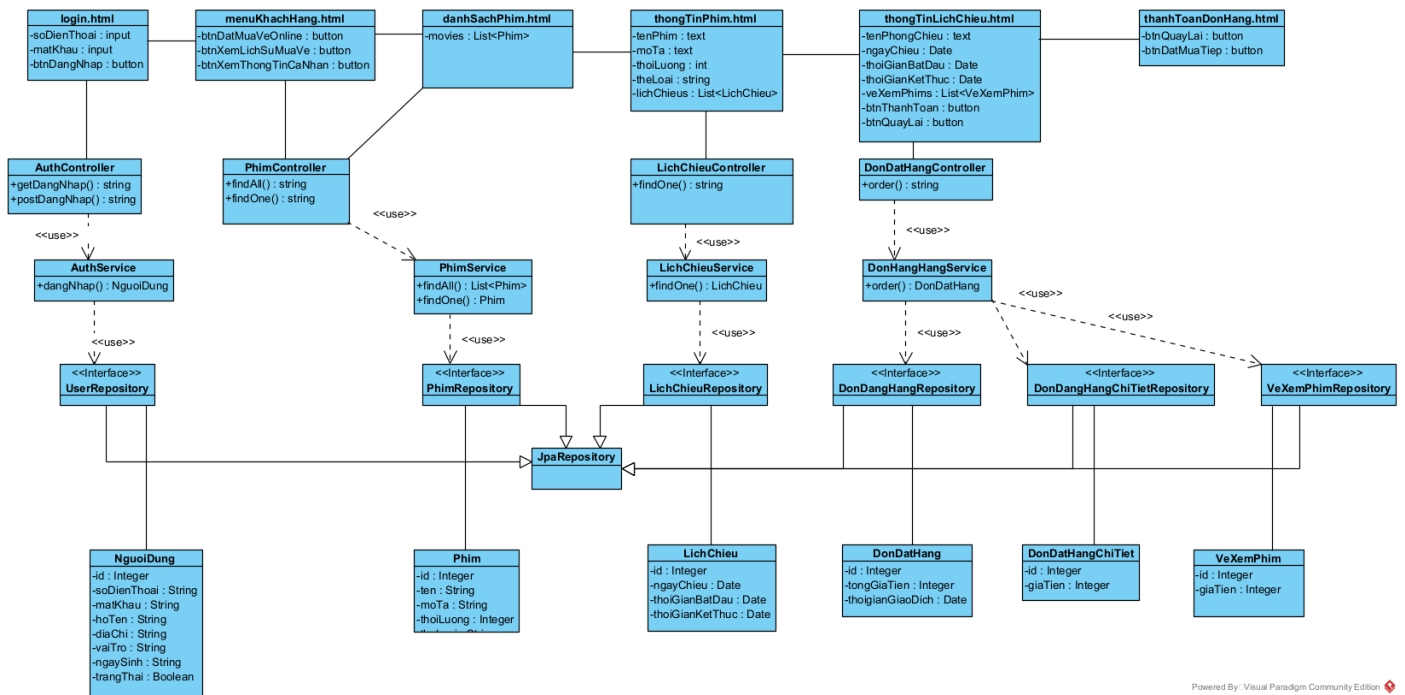
Đặt vé thành công

Cảm ơn bạn đã đặt vé! Đặt vé của bạn đã được xác nhận.

Quay về trang chủ

Đặt mua tiếp

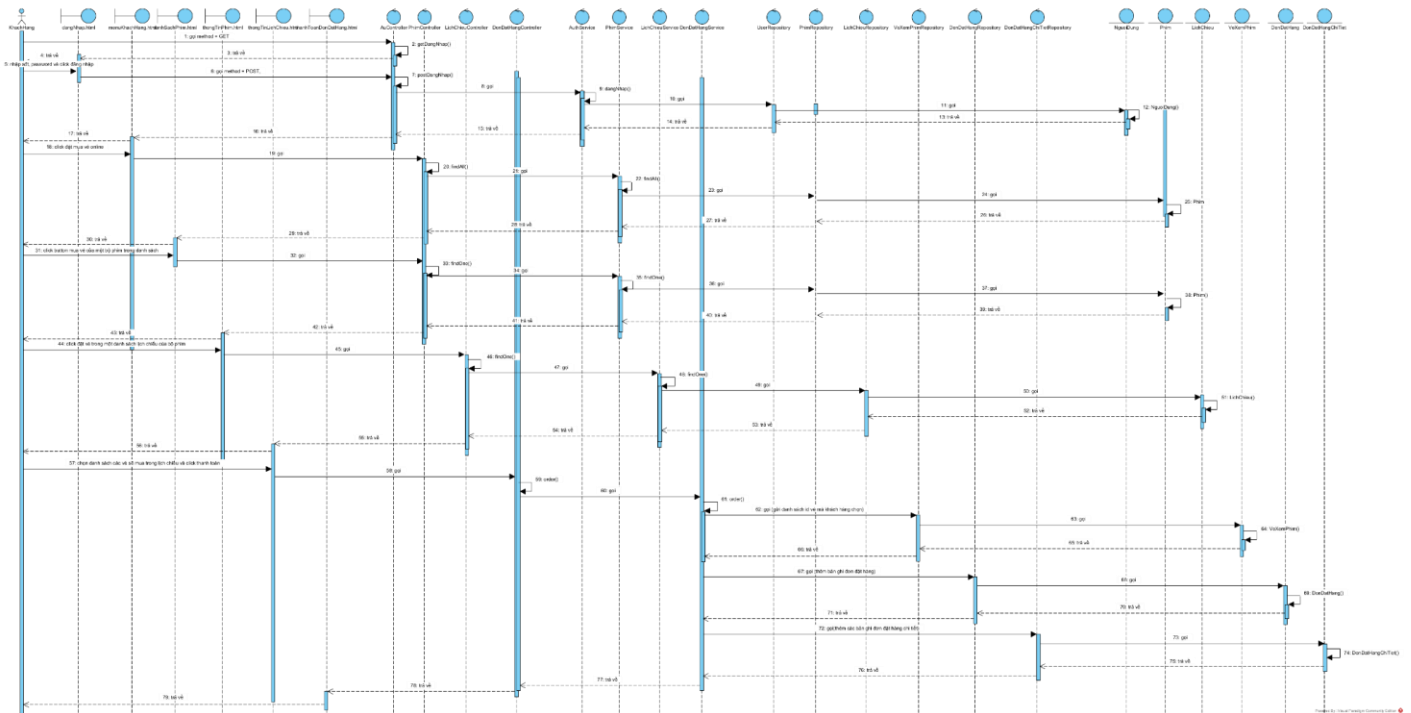
## 5.2 Thiết kế biểu đồ lớp chi tiết



- Diễn giải: Sử dụng mô hình MVC, chia thành các tầng view, model, controller và xử lý lần lượt từng tầng theo thứ tự từ trên xuống dưới
  - Cần xây dựng các giao diện HTML tương ứng với phần trên: login.html, menuKhachHang.html, danhSachPhim.html, thongTinPhim.html, thongTinLichChieu.html, thanhToanDonHang.html
  - Tạo lớp **AuthController** để nhận các request đăng nhập từ người dùng, lớp này inject AuthService, với các method getDangNhap(), postDangNhap nhận vào soDienThoai, matKhau. Lớp này inject lớp AuthService để xử lý các logic
  - Tạo lớp **PhimController** để nhận các request từ người dùng khi người dùng chọn xem danh sách các phim *method findAll()*, và khi click chi tiết thông tin 1 bộ phim *method findOne()* nhận vào là id của phim. Lớp này inject lớp PhimService để xử lý các logic cho các method trên
  - Tạo lớp **LichChieuController** để nhận các request từ người dùng khi click chi tiết vào một lịch chiếu với 1 phim cụ thể *method findOne()* nhận vào là id của lịch chiếu. Lớp này inject lớp LichChieuService để xử lý logic
  - Tạo lớp **DonDatHangController** để nhận các request từ người dùng khi chọn các vé xem phim và click đặt mua => *method order()*

- nhận vào là `List<Integer> selectedTickets`(1 mảng danh sách id của các vé xem phim đã chọn). Lớp này inject lớp `DonDatHangService`
- Tạo lớp **AuthService** có inject `UserRepository`(kế thừa `JpaRepository`) để xử lý các yêu cầu đăng nhập từ tầng `AuthController`, với method `dangNhap()` nhận 2 tham số `username`, `password` và trả về đăng nhập thành công hay không thành công
  - Tạo lớp **PhimService** có inject `PhimRepository`(kế thừa `JpaRepository`) để xử lý các yêu cầu lấy danh sách phim và lấy chi tiết một bộ phim từ tầng `PhimController` qua 2 method `findAll()` và `findOne()`
  - Tạo lớp **LichChieuService** có inject `LichChieuRepository` để xử lý yêu cầu lấy thông tin chi tiết của một lịch chiếu từ `LichChieuController` qua method `findOne()`
  - Tạo lớp **DonDatHangService** để xử lý yêu cầu đặt hàng từ `DonDatHangController` qua method `order()`.
    - Lớp này inject `DonDatHangRepository` dùng để thêm 1 bản ghi mới vào bảng `DonDatHang`,
    - inject `DonDatHangChiTietRepository` dùng để thêm danh sách các bản ghi vào bảng `DonDatHangChiTietRepository`
    - inject `VeXemPhimRepository` dùng để lấy danh sách thông tin chi tiết các vé xem phim từ danh sách ids các vé xem phim
  - Tạo interface **UserRepository** để thực hiện các truy vấn liên quan tới `NgoiDung`, tương tự với các interface còn lại...

### 5.3 Thiết kế biểu đồ tuần tự



## Diễn giải:

1. Khách hàng truy cập URL, gửi yêu cầu GET tới AuthController, method = GET
2. Lớp AuthController nhận request và thực hiện hàm GetDangNhap()
3. Hàm getDangNhap() trả về dangNhap.html
4. dangNhap.html hiển thị giao diện đăng nhập cho khách hàng
5. Khách hàng nhập số điện thoại và mật khẩu, click đăng nhập
6. dangNhap.html gửi yêu cầu tới AuthController, method = POST
7. AuthController nhận request và thực hiện hàm postDangNhap()
8. Hàm postDangNhap() của lớp AuthController gọi tới lớp AuthService
9. Lớp AuthService thực hiện hàm dangNhap()
10. Hàm dangNhap của lớp AuthService gọi tới UserRepository
11. UserRepository gọi tới lớp NguoIDung
12. Lớp NguoIDung tự đóng gói đối tượng
13. Lớp NguoIDung trả về đối tượng cho UserRepository
14. UserRepository trả về đối tượng cho lớp AuthService
15. Lớp AuthService trả về đối tượng cho lớp AuthController
16. Lớp AuthController trả về menuKhachHang.html
17. menuKhachHang.html hiện lên giao diện khách hàng
18. Khách hàng click button đặt mua vé online
19. menuKhachHang.html gửi request tới lớp PhimController, method = GET, endpoint = /movie/all

20. Lớp PhimController thực hiện hàm findAll()
21. Hàm findAll() của lớp PhimController gọi tới PhimService
22. Lớp PhimService thực hiện hàm findAll()
23. Hàm findAll() của lớp PhimService gọi tới PhimRepository
24. PhimRepository gọi tới lớp Phim
25. Lớp Phim tự đóng gói đối tượng
26. Lớp Phim trả về đối tượng cho PhimRepository
27. PhimRepository trả về đối tượng cho PhimService
28. Lớp PhimService trả về đối tượng cho PhimController
29. PhimController trả về danhSachPhim.html
30. danhSachPhim.html hiện lên giao diện danh sách các bộ phim đang được bán
31. Khách hàng click vào button mua vé của một bộ phim trong số danh sách
32. danhSachPhim.html gửi request tới PhimController, method = GET, endpoint = /movie/id
33. Lớp PhimController thực hiện hàm findOne()
34. Hàm findOne() của lớp PhimController gọi tới lớp PhimService
35. Lớp PhimService thực hiện hàm findOne()
36. Hàm findOne() của lớp PhimService gọi tới PhimRepository
37. PhimRepository gọi tới lớp Phim
38. Lớp Phim tự đóng gói đối tượng
39. Lớp Phim trả về đối tượng cho PhimRepository
40. PhimRepository trả về đối tượng cho PhimService
41. Lớp PhimService trả về đối tượng cho PhimController
42. Lớp PhimController trả về thôngTinPhim.html
43. thôngTinPhim.html hiện lên giao diện thông tin chi tiết 1 bộ phim cụ thể và danh sách các lịch chiếu của bộ phim đấy
44. Khách hàng click vào button đặt vé của một lịch chiếu cụ thể trong các danh sách lịch chiếu
45. thôngTinPhim.html gửi request tới LichChieuController, method = GET, endpoint = /screening/id
46. Lớp LichChieuController nhận request và thực hiện hàm findOne()
47. Hàm findOne của lớp LichChieuController gọi tới lớp LichChieuService
48. Lớp LichChieuService thực hiện hàm findOne()
49. Hàm findOne() của lớp LichChieuService gọi tới LichChieuRepository

50. LichChieuRepository tự thực hiện hàm ***findOneByld(lấy thông tin chi tiết bản ghi lịch chiếu hiện tại theo id, đồng thời JOIN FETCH các vé xem phim của lịch chiếu đấy)***, gọi tới lớp LichChieu
51. Lớp LichChieu tự đóng gói đối tượng
52. Lớp LichChieu trả về đối tượng cho LichChieuRepository
53. LichChieuRepository trả về đối tượng cho LichChieuService
54. Lớp LichChieuService trả về đối tượng cho LichChieuController
55. Lớp LichChieuController trả về thôngTinLichChieu.html
56. thôngTinLichChieu.html hiện lên giao diện thông tin chi tiết của một lịch chiếu(gồm thông tin phòng chiếu, ngày chiếu, thời gian bắt đầu và kết thúc lịch chiếu) và danh sách các vé của lịch chiếu đấy
57. Khách hàng chọn 1 hoặc nhiều các vé của lịch chiếu đấy và click button thanh toán
58. thôngTinLichChieu.html gửi request tới DonDatHangController, method = POST, endpoint = /order, body(danh sách id của các vé)
59. DonDatHangController nhận request và thực hiện hàm order()
60. Hàm order() của DonDatHangController gọi tới lớp DonDatHangService
61. Lớp DonDatHangService thực hiện hàm order()
62. Hàm order() của lớp DonDatHangService gọi tới VeXemPhimRepository
63. VeXemPhimRepository tự thực hiện hàm ***findAllBylds(lấy toàn bộ các vé xem phim theo các ids)***, gọi tới lớp VeXemPhim
64. Lớp VeXemPhim tự đóng gói đối tượng
65. Lớp VeXemPhim trả về đối tượng cho VeXemPhimRepository
66. VeXemPhimRepository trả về đối tượng cho DonDatHangService
67. Hàm order() của lớp DonDatHangService gọi tới DonDatHangRepository
68. DonDatHangRepository tự thực hiện hàm ***save(thêm 1 bản ghi đơn đặt hàng)***, gọi tới lớp DonDatHang
69. Lớp DonDatHang tự đóng gói đối tượng
70. Lớp DonDatHang trả về đối tượng cho DonDatHangRepository
71. DonDatHangRepository trả về đối tượng cho DonDatHangService
72. Hàm order() của lớp DonDatHangService gọi tới DonDatHangChiTietRepository
73. DonDatHangChiTietRepository tự thực hiện hàm ***saveAll(thêm danh sách các bản ghi đơn đặt hàng chi tiết)***, gọi tới lớp DonDatHangChiTiet
74. Lớp DonDatHangChiTiet tự đóng gói đối tượng
75. Lớp DonDatHangChiTiet trả về đối tượng cho DonDatHangChiTietRepository

- 76. DonDatHangChiTietRepository trả về đối tượng cho DonDatHangService
- 77. Lớp DonDatHangService kết thúc hàm order() trả về đối tượng cho DonDatHangController
- 78. Lớp DonDatHangController trả về thanhToanDonHang.html
- 79. thanhToanDonHang.html hiện lên giao diện thanh toán thành công và button quay về trang chủ, tiếp tục đặt vé online

## 6. Module Khách hàng mua vé online

### 6.1 Thiết kế giao diện

#### - **Giao diện đăng nhập**

Login

Số điện thoại

0987838802

Mật khẩu

Đăng nhập

#### - **Giao diện menu quản lý**

Hệ thống quản lý phim, MANAGER Nguyễn Đức Anh!

Quản lý thông tin khách hàng

Xem Thông Kê Khách Hàng

Xem Thông Tin Cá Nhân

#### - **Giao diện xem thống kê danh sách khách hàng:**

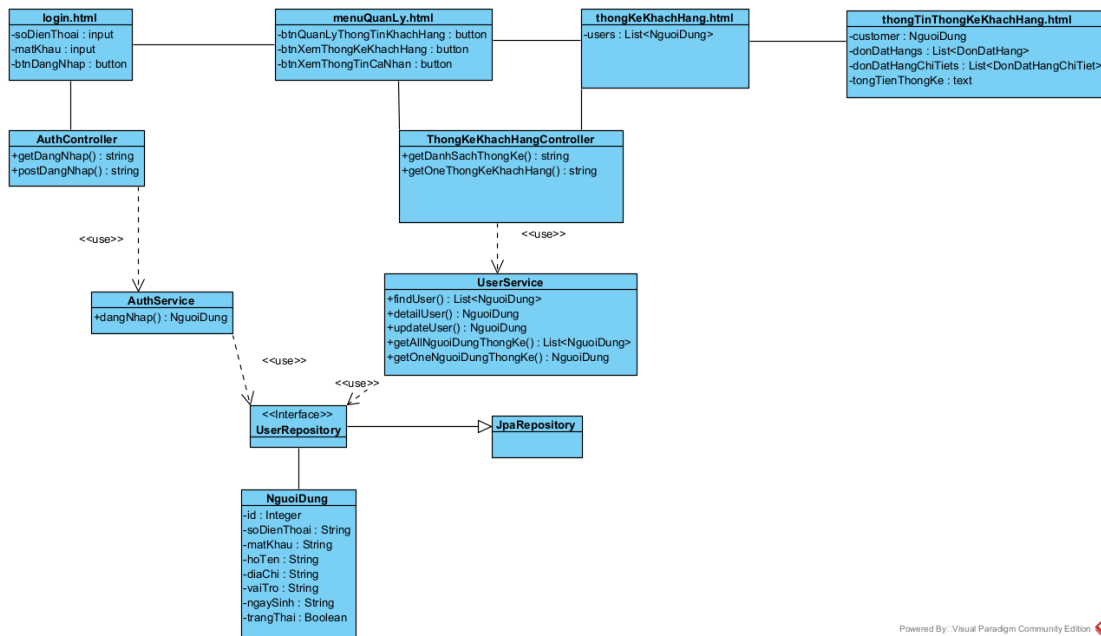
Danh sách khách hàng và số tiền đã chi
<div><div>Nguyễn Đức Anh(Customer)</div><div>Số tiền đã chi: 100000</div><div>Xem chi tiết</div></div>
<div><div>Vũ Thị Thủy</div><div>Số tiền đã chi: 50000</div><div>Xem chi tiết</div></div>

- ***Giao diện xem thông kê chi tiết của một khách hàng:***

Thông Tin Khách Hàng
<div><div>Nguyễn Đức Anh(Customer)</div><div>Địa chỉ: Hà Nội</div><div>Số điện thoại: 0987838801</div></div>
<div><div>Danh sách đơn mua vé</div><div><div>Đơn số 0 - 2024-04-14 21:39:14.562</div><div><div>Tổng giá tiền: 100000</div><div>Chi tiết vé:</div><div><div>Ghế: A10, Giá: 50000</div><div>Ghế: A12, Giá: 50000</div></div></div></div></div>

6.2 Thiết kế biểu đồ lớp chi tiết





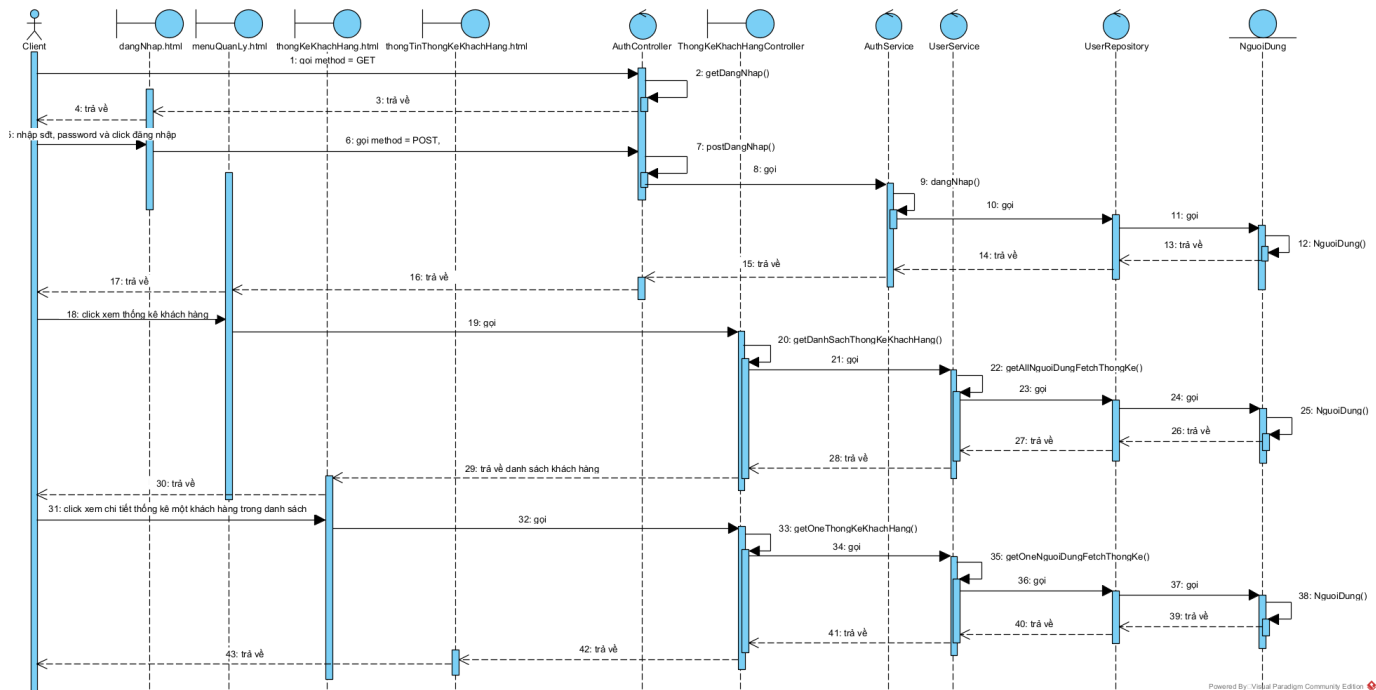
- Diễn giải: Sử dụng mô hình MVC, chia thành các tầng view, model, controller và xử lý lần lượt từng tầng theo thứ tự từ trên xuống dưới
  - Cần xây dựng các giao diện HTML tương ứng với phần trên: login.html, menuQuanLy.html, thongKeKhachHang.html, thongTinThongKeKhachHang.html
  - Tạo lớp **AuthController** để nhận các request đăng nhập từ người dùng, lớp này inject AuthService, với các method getDangNhap(), postDangNhap nhận vào soDienThoai, matKhai. Lớp này inject lớp AuthService để xử lý các logic
  - Tạo lớp **ThongKeKhachHangController** để nhận các request từ người dùng khi xem danh sách thống kê chi tiêu, và thông tin thống kê chi tiết của một khách hàng => method *getDanhSachThongKe()* và *getOneThongKeKhachHang()* nhận vào id khách hàng. Lớp này inject lớp UserService để xử lý logic các method trên
  - Tạo lớp **AuthService** có inject UserRepository(kế thừa JpaRepository) để xử lý các yêu cầu đăng nhập từ tầng AuthController, với method dangNhap() nhận 2 tham số username, password và trả về đăng nhập thành công hay không thành công
  - Tạo lớp **UserService**, lớp này xử lý các yêu cầu từ tầng ThongKeKhachHangController, ở đây chỉ diễn giải 2 method *getAllThongKeNguoiDung()* và *getOneThongKeNguoiDung()*, các method còn lại là của module quản lý thông tin khách hàng. Lớp này inject UserRepository để thực hiện các truy vấn từ cơ sở dữ liệu, sẽ

truy vấn từ table `NguoIDung` và JOIN FETCH tới

`NguoIDung.DonDatHang` thông qua khóa ngoại ở bảng `DonDatHang`

- Tạo interface **UserRepository** để thực hiện các truy vấn liên quan tới `NguoIDung` => truy vấn find thông qua `soDienThoai`, truy vấn `findAll` và join với `DonDatHang`, ...

### 6.3 Thiết kế biểu đồ tuần tự



#### Diễn giải:

1. Quản lý truy cập URL, gửi yêu cầu tới `AuthController`, method = GET
2. Lớp `AuthController` nhận request và thực hiện hàm `getDangNhap()`
3. Hàm `getDangNhap()` trả về `dangNhap.html`
4. `dangNhap.html` hiện lên giao diện đăng nhập quản lý
5. Quản lý nhập đúng số điện thoại và mật khẩu và click đăng nhập
6. `dangNhap.html` gửi yêu cầu tới `AuthController` method = POST
7. `AuthController` nhận request và thực hiện hàm `postDangNhap()`
8. Hàm `postDangNhap()` của `AuthController` gọi tới lớp `AuthService`
9. Lớp `AuthService` thực hiện hàm `dangNhap()`
10. Hàm `dangNhap()` của lớp `AuthService` gọi tới `UserRepository`
11. `UserRepository` gọi tới lớp `NguoIDung`
12. Lớp `NguoIDung` tự đóng gói đối tượng

13. Lớp `NguoiDung` trả về đối tượng cho `UserRepository`
14. `UserRepository` trả về đối tượng cho `AuthService`
15. Lớp `AuthService` trả về đối tượng cho `AuthController`
16. Lớp `AuthController` trả về `menuQuanLy.html`
17. `menuQuanLy.html` hiện lên giao diện menu của quản lý
18. Quản lý click vào nút xem thống kê khách hàng
19. `menuQuanLy.html` gửi request tới lớp `ThongKeKhachHangController`,  
method = GET, endpoint = `/thongKeKhachHang/all`
20. Lớp `ThongKeKhachHangController` thực hiện hàm  
`getDanhSachThongKeKhachHang()`
21. Hàm `getDanhSachThongKeKhachHang()` của lớp  
`ThongKeKhachHangController` gọi tới lớp `UserService`
22. Lớp `UserService` thực hiện hàm `getAllNguoiDungThongKe()`
23. Hàm `getAllNguoiDungThongKe()` của lớp `UserService` gọi tới `UserRepository`
24. `UserRepository` tự thực hiện hàm **`findAll(có JOIN FETCH các lớp DonDatHang)`**, gọi tới lớp `NguoiDung`
25. Lớp `NguoiDung` tự đóng gói đối tượng
26. Lớp `NguoiDung` trả về đối tượng cho `UserRepository`
27. `UserRepository` trả về đối tượng cho lớp `UserService`
28. Lớp `UserService` trả về đối tượng cho `ThongKeKhachHangController`
29. Lớp `ThongKeKhachHangController` trả về `thongKeKhachHang.html`
30. `thongKeKhachHang.html` hiện lên giao diện danh sách người dùng, và số tiền  
đã sử dụng của mỗi người dùng
31. Quản lý click vào nút xem chi tiết thống kê của một người dùng cụ thể
32. `thongKeKhachHang.html` gửi request tới `ThongKeKhachHangController`,  
method = GET, endpoint = `/thongKeKhachHang/id`
33. Lớp `ThongKeKhachHangController` thực hiện hàm  
`getOneThongKeKhachHang()`
34. Hàm `getOneThongKeKhachHang` của lớp `ThongKeKhachHangController` gọi  
tới `UserService`
35. Lớp `UserService` thực hiện hàm `getOneNguoiDungFetchThongKe()`
36. Hàm `getOneNguoiDungFetchThongKe` gọi tới `UserRepository`
37. `UserRepository` tự thực hiện hàm **`findOne(có JOIN FETCH các lớp DonDatHang, DonDatHangChiTiet)`**, gọi tới lớp `NguoiDung`
38. Lớp `NguoiDung` tự đóng gói đối tượng
39. Lớp `NguoiDung` trả về đối tượng cho `UserRepository`
40. `UserRepository` trả về đối tượng cho lớp `UserService`

41. Lớp UserService trả về đối tượng cho ThôngKeKhachHangController
42. Lớp ThôngKeKhachHangController trả về thôngTinThôngKeKhachHang.html
43. thôngTinThôngKeKhachHang.html hiện lên giao diện thông tin cá nhân của một khách hàng cụ thể, danh sách các đơn hàng đặt vé của khách hàng đấy, với mỗi đơn hàng lại hiển thị danh sách các đơn đặt hàng chi tiết của từng vé