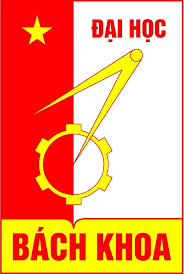
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**



BÁO CÁO BÀI TẬP LỚN

THỰC HÀNH CƠ SỞ DỮ LIỆU

PROJECT: THIẾT KẾ CƠ SỞ DỮ LIỆU SỬ DỤNG ĐỒ HỌA

Giảng viên : Nguyễn Thị Oanh

Họ và tên thành viên:

Mai Hoàng Minh – 20184151

Nguyễn Duy Khánh – 20184125

**MỤC LỤC**

1. Giới thiệu chung
2. Nội dung
3. Mô tả nghiệp vụ, chức năng
4. Nghiệp vụ
5. Chức năng
6. Thực hiện
7. Database

* Sơ đồ thực thể liên kết
* Thiết kế Database

1. Viết Source Code
2. Tổng kết
3. Phân công nhiệm vụ
4. Ưu điểm, nhược điểm
5. Ưu điểm
6. Nhược điểm
7. Phát triển trong tương lai

I. GIỚI THIỆU CHUNG

* Thiết kế một ứng dụng Quản lí khách sạn.
* Sử dụng những kiến thức đã học về Database PostgreSQL để thiết kế cơ sở dữ liệu, các câu lệnh truy vấn thực hiện các chức năng cần thiết.
* Sử dụng ngôn ngữ lập trình Java để viết thành ứng dụng, kết hợp với JDBC giúp kết nối với Database và Java Swing thiết kế đồ họa.

II. NỘI DUNG

1. Mô tả nghiệp vụ, chức năng
2. Nghiệp vụ

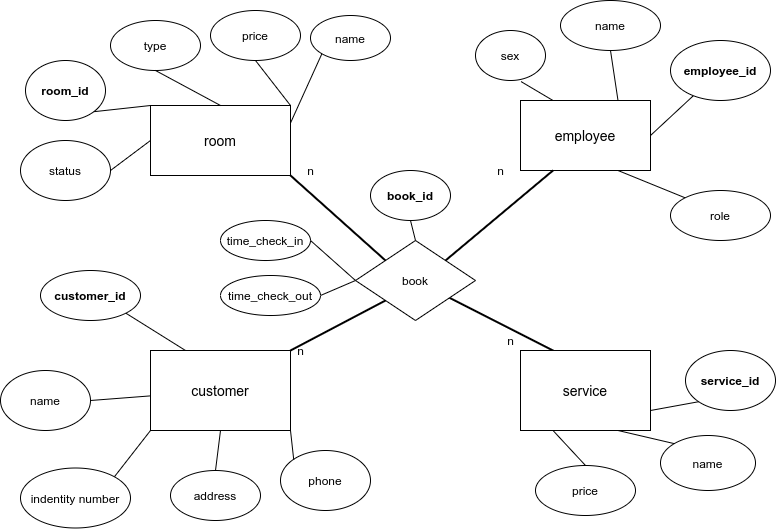
* Khách sạn gồm danh sách các nhân viên, mỗi nhân viên sẽ gồm các đặc tính : Mã nhân viên, tên, giới tính, chức vụ.
* Danh sách các phòng trong khách sạn gồm các thuộc tính : Mã phòng, tên, loại phòng, giá, tình trạng.
* Các dịch vụ có trong khách sạn sẽ được lưu, gồm các thuộc tính : Mã dịch vụ, tên, giá.
* Thông tin về khách đặt phòng gồm : Mã khách đặt phòng, tên, giới tính, số CMND, địa chỉ, số điện thoại. Khách khi đặt phòng sẽ có thông tin đặt phòng gồm : Mã hóa đơn, thời gian đặt, thời gian trả, tên phòng, tên nhân viên và các dịch vụ kèm theo.

1. Chức năng

* Người có chức vụ quản lí sẽ có khả năng xem thông tin của nhân viên, phòng và dịch vụ. Bên cạnh đó còn có thể chỉnh sửa thông tin đó như thêm, sửa, xóa, cập nhật. Người quản lí cũng xem được các thông tin liên quan đến kinh doanh của khách sạn như thu nhập, số lượng đặt phòng, khách hàng thường xuyên.
* Nhân viên sẽ trực ở sảnh khách sạn theo ca của mình. Khi có khách đến đặt phòng, nhân viên sẽ thực hiện “Book phòng” bằng cách nhập các thông tin như phòng, dịch vụ, thời gian check out và thông tin của khách hàng. Sau khi nhập thông tin hợp lệ, nhân viên có thể xuất ra hóa đơn thanh toán để trả lại khách hàng : Mã đặt phòng, ngày check in, ngày check out, tên nhân viên phụ trách quản lí phòng, số tiền phải trả.

1. Thực hiện
2. Database

* Sơ đồ thực thể liên kết :



* Thiết kế Database :

employee(**employee\_id**, name, sex, role)

room(**room\_id**, name, type, price, status)

service(**service\_id**, name, price)

customer(**customer\_id**, name, address, indentity\_number, phone)

book(**book\_id**, *customer\_id*, customer\_name, *room\_id, service\_id, employee\_id*, time\_check\_in,

time\_check\_out)

useraccount(**username**, password, role, *employee\_id*)

1. Viết Source Code

* Sử dụng ngôn ngữ lập trình Java, Database Driver JDBC, thiết kế đồ họa Java Swing.
* Thiết kế theo mô hình MVC (Model – View – Controller)
* Model

- Thực hiện các thao tác với Database, thông qua JDBC.

- Chuyển đổi giữa câu lệnh SQL và câu lệnh Java.

- Mỗi một bảng sẽ được viết thành các đối tượng “Entity”.

- Các Model Object như Data Access Object (DAO) và Data Transfer Object (DTO) sẽ thực hiện truy vấn với Database và truyền dữ liệu giữa các tầng.

* Controller

- Cầu nối giữa View và Model.

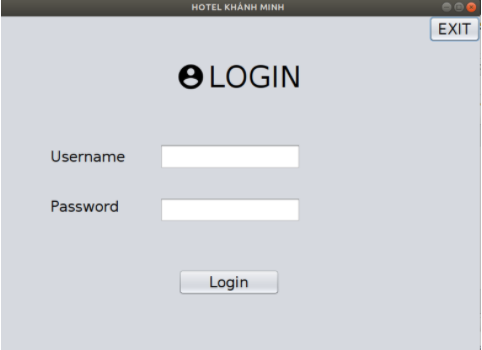
- Thực hiện các Logic, thao tác nhận từ View, tác động vào Database qua Model. Sau khi truy vấn thành công trong Database, sẽ trả lại kết quả cho người dùng trên View.

* View

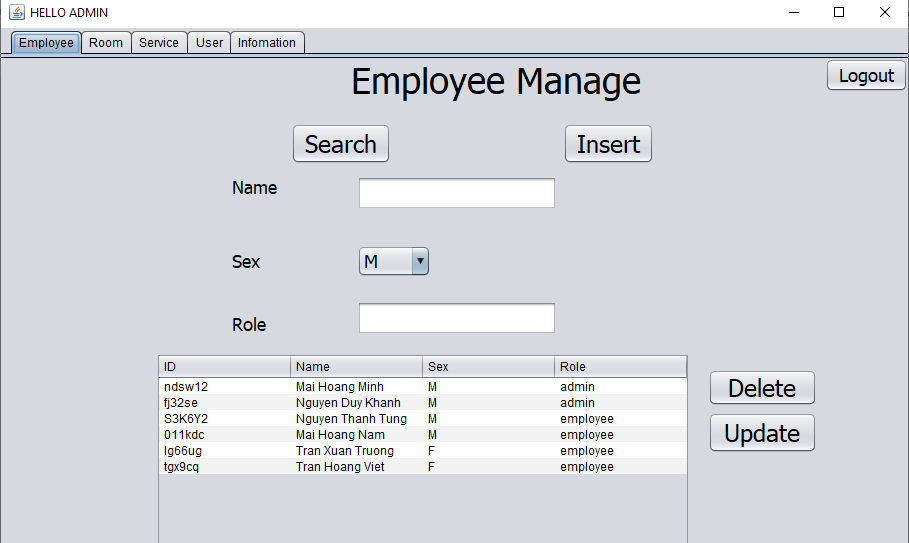
- Tương tác giữa người dùng và Ứng dụng.

- Đưa các thao tác vào bên trong để Controller xử lí.

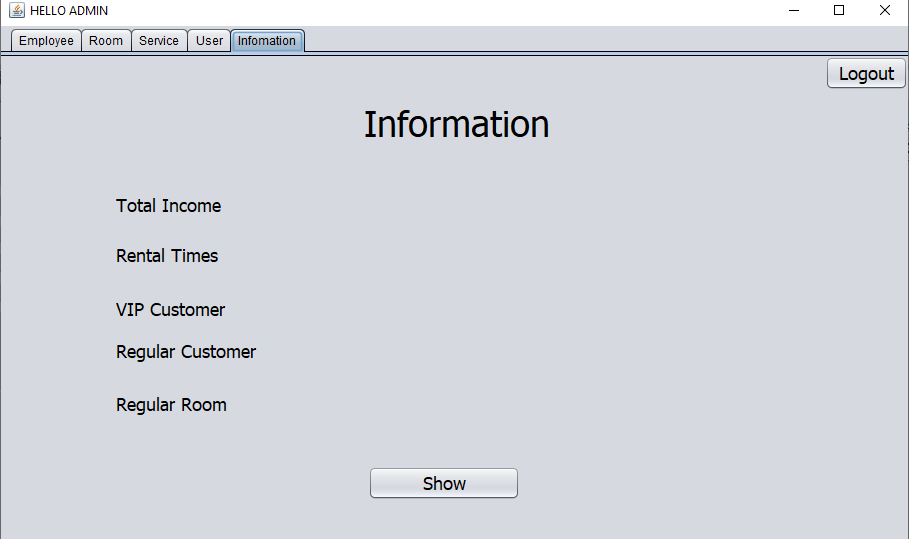
* Chức năng
* Chức năng đăng nhập



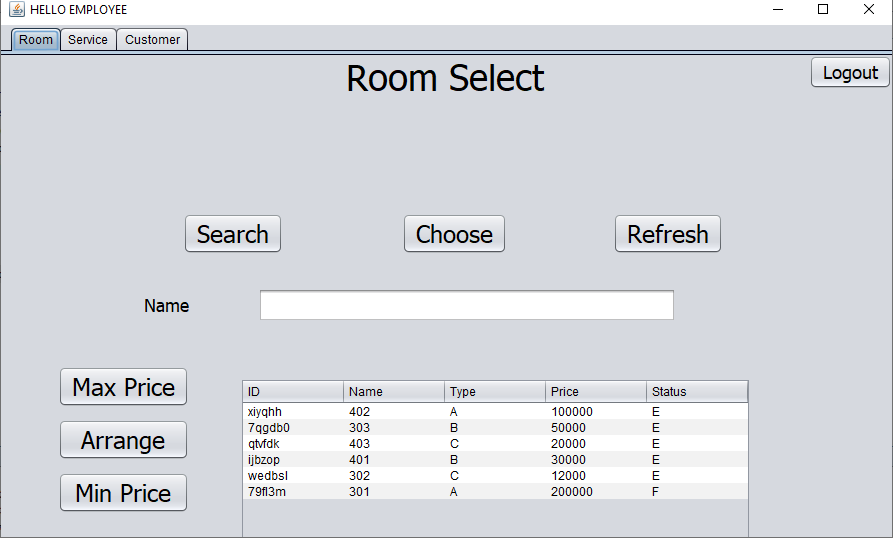
* Sử dụng màn hình đăng nhập với username và password (thông tin username và password mã hóa được lưu trữ trong database). Từ đó xác định được người đăng nhập là admin (người quản lí) hay employee (nhân viên).
* Các thông tin nhập vào luôn có check lỗi ở từng thông tin, giúp người dùng nắm bắt tốt hơn thông tin khi mình nhập vào; Mật khẩu cũng được mã hóa bằng MD5, giúp an toàn hơn, tránh trường hợp xâm phạm trái phép.
* Chức năng cho Admin (quản lí)



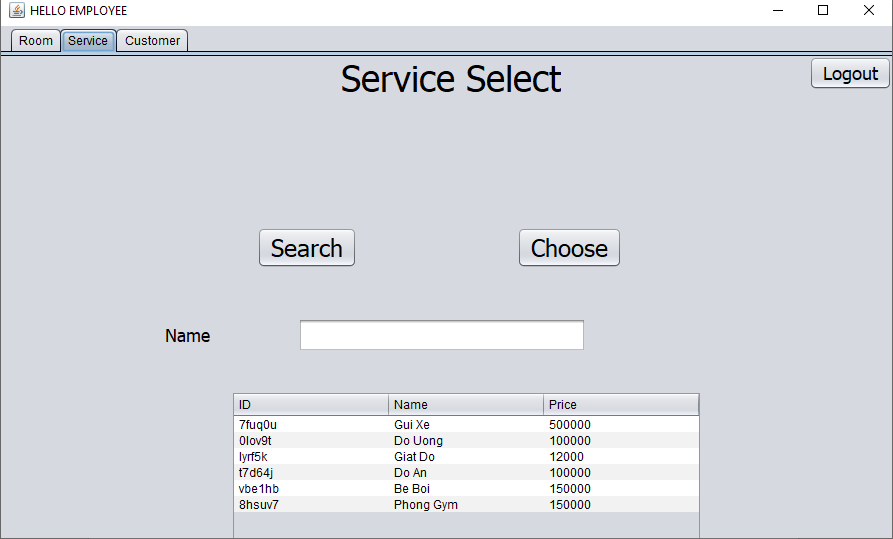
* Admin sẽ có khả năng xem, thêm, sửa, xóa thông tin Nhân viên, Phòng, Dịch vụ. Bên cạnh đó, Admin cũng có thể tác động tới thông tin User (dùng để đăng nhập vào hệ thống).
* Xem thông tin kinh doanh của khách sạn : Thu nhập (Total Income), số lượt thuê (Rental times), khách hàng thân thiết (VIP Customer, Regular Customer), phòng “hot” (Regular Room).

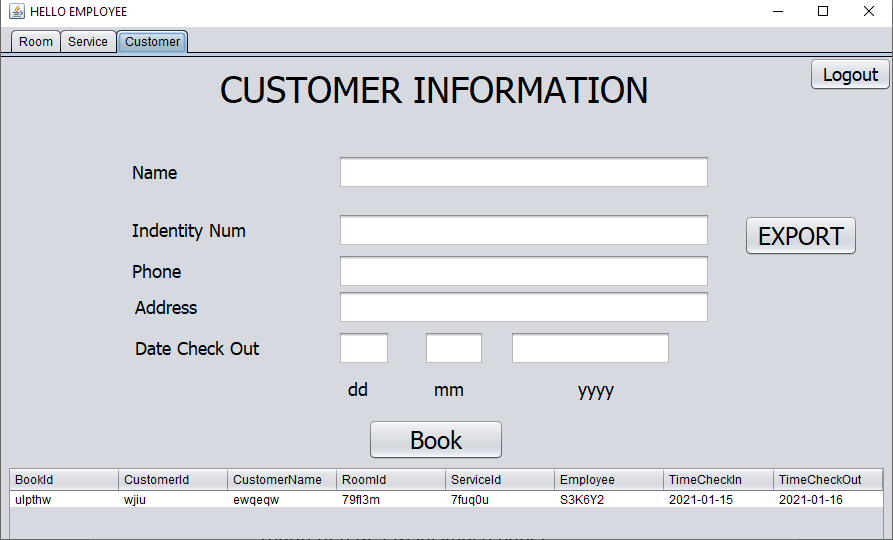


* Từ phiên đăng nhập của Admin, có thể Log Out ra được màn hình đăng nhập.
* Chức năng Employee (nhân viên)
* Chức năng chính của nhân viên là thực hiện việc đặt phòng.
* Khi khách tới đặt phòng, nhân viên sẽ trải qua các bước đặt phòng lần lượt là : Chọn phòng -> Chọn dịch vụ -> Nhập thông tin đặt phòng -> Xuất hóa đơn.

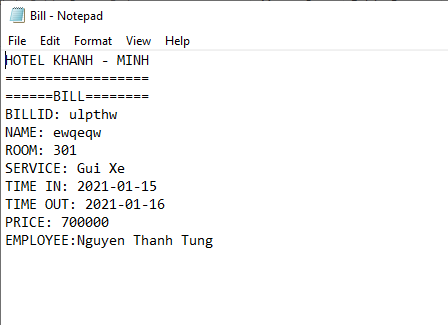


* Chức năng Refresh : những khách đã trả phòng thì nhân viên sẽ dùng chức năng này để chuyển trạng thái từ F thành E cho phòng đã trả tương ứng.





* Mỗi bước sẽ được lập trình các thao tác tìm phòng (theo giá, theo thứ tự), tìm dịch vụ tự động, check khi thông tin khách hàng nhập vào.
* Hóa đơn xuất ra file text sẽ được tính tự động thành tiền để gửi lại khách hàng.



III. TỔNG KẾT

1. Phân công nhiệm vụ

|  |  |  |
| --- | --- | --- |
| Nhiệm vụ | Mai Hoàng Minh | Nguyễn Duy Khánh |
| Ý tưởng | 70% | 30% |
| Thiết kế Database | 50% | 50% |
| Các câu lệnh truy vấn | 50% | 50% |
| Code Java | 90% | 10% |
| Code đồ họa | 40% | 60% |
| Check, sửa lỗi | 30% | 70% |
| **Tổng** | 50% | 50% |

**Câu Lệnh Truy Vấn :**

* Nguyễn Duy Khánh

------------------------------

create trigger af\_insert\_book

after insert on book

for each row

execute procedure af\_insert\_book\_func();

------------------------------

create or replace function af\_insert\_book\_func() returns trigger as $$

begin

update book set time\_check\_in = current\_date;

return new;

end;

$$ language plpgsql;

------------------------------

create or replace function check\_time\_out\_func(in val date) returns int as

$$ declare bookvar book%rowtype;

begin

for bookvar in select \* from book loop

if val > bookvar.time\_check\_out

then

update room set status = 'E' where room\_id in (select room\_id

from book

where time\_check\_out = bookvar.time\_check\_out);

end if;

end loop;

return 5;

end;

$$ language plpgsql;

------------------------------

SELECT check\_time\_out\_func(current\_date);

------------------------------

SELECT COUNT(\*) FROM book;

------------------------------

SELECT \* FROM book WHERE name LIKE '%?%';

------------------------------

INSERT INTO book(book\_id, customer\_id, customer\_name, room\_id, service\_id, employee\_id, time\_check\_out)

VALUES (?, ?, ?, ?, ?, ?, ?);

------------------------------

SELECT \* FROM room

WHERE price <= ALL(

SELECT min(price) FROM room);

------------------------------

SELECT \* FROM room

order by price DESC;

------------------------------

SELECT \* FROM useraccount;

* Mai Hoàng Minh

------------------------------

create view bill as

select book\_id, b.customer\_name, r.name as "room\_name", sv.name as "service\_name",

e.name as "employee\_name", time\_check\_in, time\_check\_out,

(time\_check\_out::date - time\_check\_in::date) \* (sv.price + r.price) as "total\_price"

from book b, room r, service sv, employee e

where b.room\_id = r.room\_id and b.service\_id = sv.service\_id and b.employee\_id = e.employee\_id;

------------------------------

SELECT customer\_name FROM bill

group by customer\_name having count(\*) >= ALL(

SELECT count(\*) FROM bill

group by customer\_name)

LIMIT 1;

------------------------------

SELECT customer\_name FROM bill

group by customer\_name having sum(total\_price) >= ALL(

SELECT sum(total\_price) FROM bill

group by customer\_name)

LIMIT 1;

------------------------------

SELECT SUM(total\_price) FROM bill;

------------------------------

SELECT \* FROM bill WHERE book\_id = '?';

------------------------------

SELECT r.name FROM book b INNER JOIN room r ON b.room\_id = r.room\_id

group by r.name having count(r.name) >= ALL (

select count(r.name) from book b inner join room r on b.room\_id = r.room\_id

group by r.name)

LIMIT 1;

------------------------------

SELECT r.name FROM book b, room r

WHERE b.room\_id = r.room\_id

group by r.name having count(r.name) >= ALL (

SELECT count(book\_id) FROM book

group by book\_id)

LIMIT 1;

------------------------------

WITH count\_book AS (

SELECT count(book\_id) FROM book

group by book\_id

)

SELECT r.name FROM book b, room r

WHERE b.room\_id = r.room\_id

GROUP BY r.name HAVING COUNT(r.name) >= ALL (select \* FROM count\_book)

LIMIT 1;

------------------------------

select \* from service where name LIKE '%?%';

------------------------------

SELECT \* FROM room

order by price DESC;

------------------------------

SELECT \* FROM room

WHERE price >= ALL(

SELECT max(price) from room);

------------------------------

select \* from customer where name LIKE '%?%';

1. Ưu điểm, nhược điểm
2. Ưu điểm

* Áp dụng được các kiến thức về PostgreSQL, ngôn ngữ lập trình Java để viết thành một ứng dụng quản lí.
* Dễ dàng sử dụng, thao tác đơn giản.
* Tính bảo mật cho user được mã hóa khá tốt.

1. Nhược điểm

* Đồ họa còn đơn giản.

1. Phát triển trong tương lai

* Thiết kế tốt hơn về đồ họa.
* Tăng tính bảo mật cho ứng dụng.
* Thực hiện thêm nhiều chức năng phức tạp hơn (đồ thị, đánh giá).