**TRƯỜNG ĐẠI HỌC PHENIKAA   
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO BÀI TẬP LỚN**

**Học Phần: *Lập trình hướng đối tượng***

Đề tài: XÂY DỰNG ỨNG DỤNG

HỖ TRỢ TÌM NHÀ (PHÒNG) TRỌ Ở HÀ NỘI

|  |  |  |
| --- | --- | --- |
| *Giảng viên hướng dẫn* | : | TS.Trần Đăng Hoan |
| *Sinh viên thực hiện* | : | 19010042 – Phạm Ngọc Việt  19010002 – Trịnh Tuấn Anh |
|  |  | 19010041 – Đỗ Thanh Tùng |
|  |  |  |
|  |  |  |

HÀ NỘI – 201...

**MỤC LỤC**

[**I.** **LỜI NÓI ĐẦU** 4](#_Toc76547858)

[**II.** **TỔNG QUAN VỀ CHƯƠNG TRÌNH (Ứng dụng)** 5](#_Toc76547859)

[**1.** **Mô tả ý tưởng** 5](#_Toc76547860)

[**2.** **Mô tả chi tiết** 6](#_Toc76547861)

[**3.** **MVP của sản phẩm** 6](#_Toc76547862)

[**3.1.** **Ý tưởng ban đầu:** 6](#_Toc76547863)

[**3.2.** **MVP (Minimum Viable Product)** 6](#_Toc76547864)

[**4.** **User Stories** 7](#_Toc76547865)

[**4.1.** **Màn hình Đăng nhập:** 7](#_Toc76547866)

[**4.2.** **Màn hình Đăng ký** 7](#_Toc76547867)

[**4.3.** **Màn hình Chính** 7](#_Toc76547868)

[**4.4.** **Màn hình sản phẩm** 8](#_Toc76547869)

[**4.5.** **Màn hình chi tiết sản phẩm** 8](#_Toc76547870)

[**4.6.** **Màn hình Giỏ hàng** 8](#_Toc76547871)

[**4.7.** **Màn hình thông tin khách hàng** 8](#_Toc76547872)

[**4.8.** **Màn hình MAP** 8](#_Toc76547873)

[**4.9.** **Màn hình Thông tin** 8](#_Toc76547874)

[**III.** **TỔNG QUAN** 9](#_Toc76547875)

[**IV.** **PHÂN TÍCH THIẾT KẾ MỨC KIẾN TRÚC CỦA BÀI TOÁN** 10](#_Toc76547876)

[**A.** **Xác định các Actor** 10](#_Toc76547877)

[**B.** **Mô Tả các use case trong hệ thống** 12](#_Toc76547878)

[***Đăng Nhập*** 12](#_Toc76547879)

[***Đăng kí*** 13](#_Toc76547880)

[***Thuê Phòng Trọ*** 14](#_Toc76547881)

[**C.** **Kiến trúc hệ thống** 15](#_Toc76547882)

[**D.** **Biểu đồ** 17](#_Toc76547883)

[**1.** **Biểu đồ lớp** 17](#_Toc76547884)

[**2.** **Phân tích bài toán bằng biểu đồ USE-CASE** 18](#_Toc76547885)

[**3.** **Biểu đồ UML** 20](#_Toc76547886)

[**4.** **Biểu đồ quan hệ** 23](#_Toc76547887)

[**E.** **Cấu trúc dữ liệu DATABASE** 24](#_Toc76547888)

[**F.** **Xây dựng các lớp** 25](#_Toc76547889)

[**a.** **CheckConnection.java** 25](#_Toc76547890)

[**b.** **MainActive.java** 26](#_Toc76547891)

[**c.** **Các lớp chính (USER)** 27](#_Toc76547892)

[**d.** **Các lớp chính (Quản lý)** 31](#_Toc76547893)

[**e.** **Giao diện hệ thống(USER)** 35](#_Toc76547894)

[**f.** **Giao diện cho Chương trình (Quản lý)** 39](#_Toc76547895)

[**V.** **ỨNG DỤNG** 43](#_Toc76547896)

[**VI.** **KẾT LUẬN** 44](#_Toc76547897)

[**VII.** **ĐÓNG GÓP TỪNG THÀNH VIÊN** 44](#_Toc76547898)

[**VIII.** **TÀI LIỆU THAM KHẢO** 45](#_Toc76547899)

1. **LỜI NÓI ĐẦU**

Hiện nay, việc tìm phòng trọ khá mệt mỏi, nhất là ở những ngày nắng nóng, những ngày mưa rét gây bất tiện và khó khăn trong việc tìm phòng trọ,nhất là đối với sinh viên hay cả những người đi làm.

Phòng trọ Hà Nội được rất nhiều người quan tâm. Không chỉ là đối tượng sinh viên và còn là những người lao động từ những tỉnh khác đến. Nhu cầu thuê phòng trọ Hà Nội trở nên quá tải nhất là vào thời điểm nhập học của sinh viên.

Với thời kỳ xã hội phát triển như hiện nay thì mọi vấn đề khó khăn đó đều được giải quyết qua chiếc điện thoại thông minh. Tìm nhà trọ qua các app là khá tiện dụng. Chính vì vậy sử dụng App để tìm nhà trọ vừa hợp lí, lại đảm bảo mọi thông tin chuẩn xác nhất đến với người dùng. Tuy vậy, trước khi sử dụng cách này để tìm nhà trọ, đầu tiên phải hiểu App tìm nhà trọ chính là những ứng dụng cung cấp thông tin, dữ liệu,... giúp người tìm phòng có thể dễ dàng nắm bắt những thông tin cần thiết về căn phòng đó.

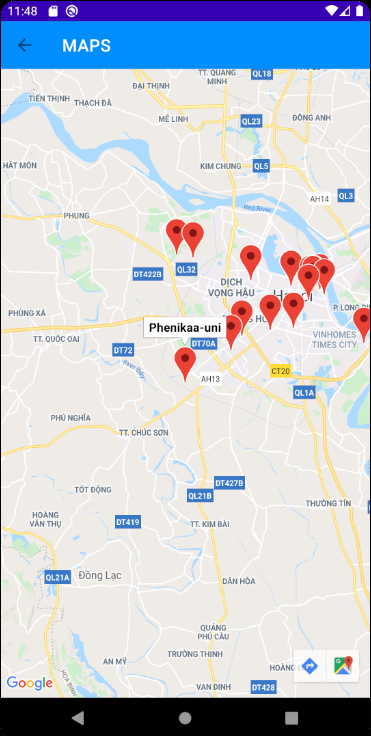
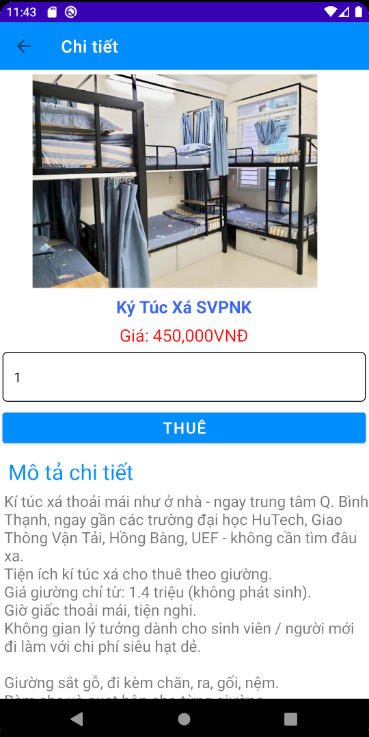
Với mục đích học tập môn lập trình hướng đối tượng tốt hơn và hoàn thành bài tập lớn chúng em đã chọn đề tài Xây dựng ứng dụng hỗ trợ tìm nhà (phòng) trọ ở Hà Nội. Đề tài này cho phép người dùng tìm kiếm những nhà trọ với giá cả hợp lý, thông tin chính xác chương trình cung cấp bản đồ và thông tin phòng trọ giúp người dùng dễ dàng tìm kiếm được các phòng trọ phù hợp mà mất ít chi phí và thời gian tìm kiếm. Chương trình được viết bởi ngôn ngữ lập trình Android - một ngôn ngữ lập trình cho ứng dụng chạy trên nền tảng mobile Android, là sự kết hợp của ngôn ngữ lập trình thuần đối tượng Java và ngôn ngữ thiết kế XML.

Do mới làm quen với ngôn ngữ Java và những hạn chế trong việc sử lý các kỹ thuật trong lập trình hướng đối tượng nên trong chương trình còn nhiều hạn chế. Nhóm chúng em mong nhận được các nhận xét của thầy về đề tài này để có thể khắc phục và làm tốt hơn trong những đề tài tiếp theo.

1. **TỔNG QUAN VỀ CHƯƠNG TRÌNH (Ứng dụng)**
2. **Mô tả ý tưởng**

App sẽ lưu trữ vị trí các khu nhà trọ, số lượng phòng, giá phòng, các trạm xe bus lân cận, hệ thống an ninh và số điện thoại liên lạc của chủ phòng trọ.

Có thể định vị bằng gps và thống kê các nhà trọ gần vị trí của người dùng.

App cập nhật giá cả các phòng trọ, số lượng một cách liên tục.

App cung cấp hình ảnh phòng trọ, diện tích...

App hỗ trợ các mô tả người dùng có thể xem xét thiết kế căn phòng có tiện hay không. Toilet có sạch sẽ hay không. Phòng trọ có thoáng mát hay đầy đủ ánh sáng không…

1. **Mô tả chi tiết**

**Mục tiêu**: tạo ra app bản đồ + thông tin phòng trọ giúp người dùng dễ dàng tìm kiếm được các phòng trọ phù hợp mà mất ít chi phí và thời gian tìm kiếm.

**Chức năng**:

* 1. Hiển thị vị trí các phòng trọ trên bản đồ và thông tin các phòng trọ đó.
  2. Thêm, sửa, xóa được các phòng trọ đó khi người quản lý có kết nối internet.
  3. Liệt kê vị trí các phòng trọ xung quanh vị trí của người dùng.
  4. Hỗ trợ cung cấp cho người dùng, bạn muốn ghép phòng chung (Nếu muốn ở nhiều hơn 1 người)

1. **MVP của sản phẩm**
   1. **Ý tưởng ban đầu:**

\* Hiển thị vị trí các phòng trọ trên bản đồ và thông tin các phòng trọ đó.

\* Thêm , sửa , xóa được các phòng trọ đó khi người dùng có thể kết nối internet.

\* Liệt kê vị trí các phòng trọ xung quanh vị trí của người dùng.

\* Hỗ trợ cung cấp tư vấn ghép phòng chung (Nếu muốn ở nhiều hơn 1 người).

* 1. **MVP (Minimum Viable Product)**

**1 - Màn hình đăng nhập**

Cho phép người dùng đăng ký và đăng nhập vào app. (có thể liên kết tài khoản Google – Facebook )

**2 - Hiển thị các phòng trọ (mới) ra màn hình chính**

Đưa ra các lựa chọn ban đầu cho người dùng.

**3 - Các mục trong Nhà trọ giá rẻ**

Hiển thị danh sách các nhà trọ đang còn trống với mức giá, địa chỉ liên hệ và mô tả về các phòng trọ.

**4 - Tính toán số lượng và giá cả**

Tổng hợp lựa chọn ghi lại và thông báo cho khách hàng số tiền cần phải trả cho những phòng đã chọn.

**5 - Liên kết bản đồ**

Hiển thị các phòng trọ trên bản đồ. Các phòng trọ xung quanh.

**6 - Hiển thị quảng cáo ở màn hình chính**

Đưa quảng cáo ra màn hình chính

1. **User Stories**
   1. **Màn hình Đăng nhập:**

Tại đây là giao diện màn hình đầu tiên mà người dùng thấy khi mở ứng dụng lên. Ở giao diện này người dùng có thể tiến hành đăng nhập tài khoản của mình để bắt đầu sử dụng ứng dụng.

* 1. **Màn hình Đăng ký**

Nếu người dùng chưa có tài khoản để đăng nhập đây là giao diện để người dùng đăng ký một tài khoản chứa thông tin cơ bản của họ. Gồm Tài khoản, Mật khẩu và địa chỉ Email. Ngoài ra người dùng có thể đăng ký bằng tài khoản Google hoặc Facebook.

* 1. **Màn hình Chính**

Đây là màn hình giao diện khi vào app. Ở đây là màn hình chính nơi để các phần như: Menu, Giỏ hàng, Logout, quảng cáo, những sản phẩm (phòng) mới nhất và hot nhất được đề xuất lên trang chính.

* 1. **Màn hình sản phẩm**

Đây là màn hình hiển thị các phòng theo từng loại đã chọn ở thanh menu. Người dùng có thể xem qua về Tên, ảnh, giá phòng.

* 1. **Màn hình chi tiết sản phẩm**

Đây là màn hình hiển thị chi tiết phòng mà người dùng lựa chọn. Tại đây có thể xem ảnh, giá, tên và Mô tả chi tiết về căn phòng được chọn.

* 1. **Màn hình Giỏ hàng**

Đây là nơi để hiển thị các phòng mà người dùng đã chọn có thể thêm hoặc bớt phòng. Có thể xóa các phòng mới chọn. Hỗ trợ tính tổng số tiền cần trả cho Đơn hàng đã chọn.

* 1. **Màn hình thông tin khách hàng**

Đây là nơi người dùng xác nhận lại phương thức liên hệ của mình với Admin khi đã quyết định thuê những phòng đã được chọn và đồng ý với mức giá đã báo.

* 1. **Màn hình MAP**

Đây là nơi hiển thị các phòng (Nhà) trọ những địa điểm có trong dữ liệu trên MAP. Có thể hiện thị tên, địa điểm. Hỗ trợ người dùng Qua ứng dụng MAP được cài trên điện thoại thông minh.

* 1. **Màn hình Thông tin**

Đây là màn hình hiển thị thông tin liên hệ với Admin. Chứa Địa chỉ, số điện thoại, Email, ....

1. **TỔNG QUAN**

**Kiến thức nền tảng phục vụ cho bài toán:**

* Phân tích và thiết kế hướng đối tượng,bao gồm :

+ Tư duy lập trình hướng đối tượng.

+ Các nguyên lý cơ bản trong lập trình hướng đối tượng (đóng gói, kế thừa, sử dụng các giao diện, đa hình, chồng phương thức...)

+ Biết sử dụng ngôn ngữ lập trình java.

+ Tìm hiểu cách thức lập trình ứng dụng bằng phương pháp hướng đối tượng

+ Kiến thức cơ bản về cơ sở dữ liệu và thiết kế cơ sở dữ liệu cho ứng dụng

1. **PHÂN TÍCH THIẾT KẾ MỨC KIẾN TRÚC CỦA BÀI TOÁN**

**Yêu cầu:**

Ứng dụng cho phép người dùng đăng nhập, đăng ký lựa chọn phòng theo nhu cầu, tài chính và mục đích sử dụng. Yêu cầu tối thiểu là ứng dụng cần tạo ra một chương trình phần mềm bao gồm:

* Thứ nhất, cho phép người dùng (người quản lý) thêm, sửa, xóa phòng trọ (nhà trọ). Với mỗi chức năng sẽ tự động cập nhật dữ liệu mới được truyền vào.
* Thứ hai, cho phép người dùng sử dụng các chức năng lựa chọn, thêm giỏ hàng, xóa khỏi giỏ hàng, tính toán giá thành tương ứng với số phòng đã đặt .
  + 1. **Xác định các Actor**

1. Khách Hàng

Là người dùng hệ thống để tìm được một nơi trọ ưng ý với tài chính cũng nhưng ý muốn của họ, được thanh toán bằng tiền mặt hoặc credit card.

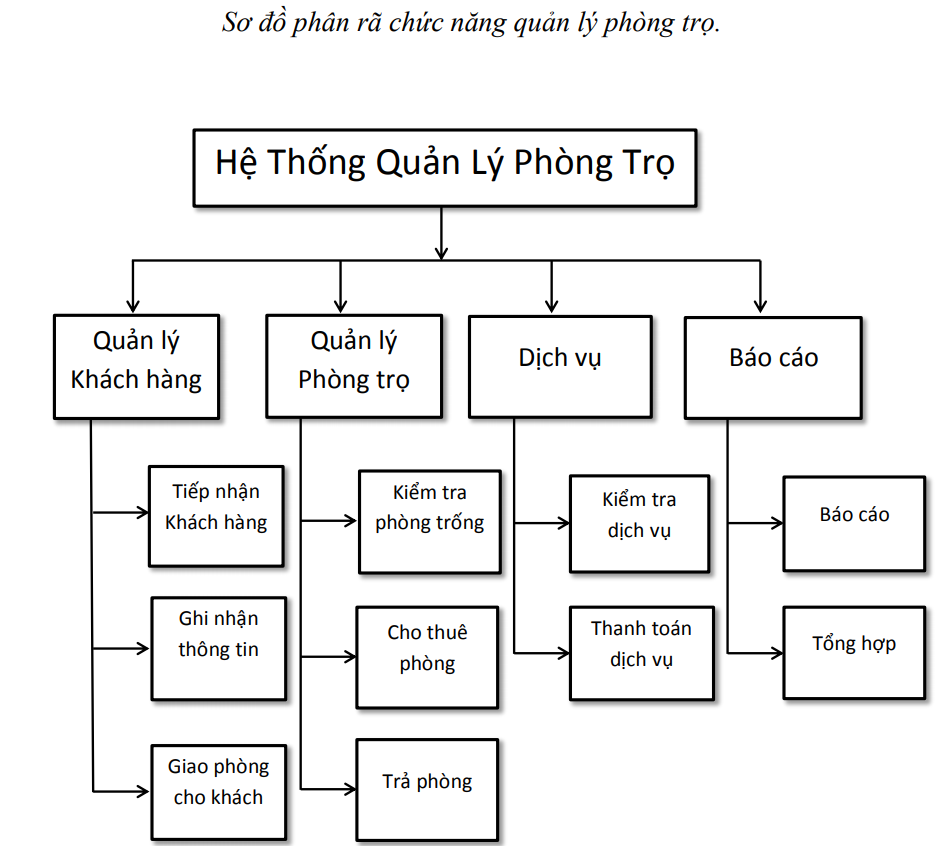
Khách hàng có tránh nhiệm phản hồi với người quản lí về những sai sót cũng nhưng lỗi của hệ thống để người quản lí có thể kịp thời xử lí hoặc đưa ra các giải pháp phù hợp để không làm ảnh hưởng đến quyền lợi cũng nhưng trải nghiệm của khách hàng.

1. Người quản lí

Là người nắm giữ tình hình hoạt động cũng như giao dịch của hệ thống.

Thống kê doanh số, chất lượng của các trọ trong hệ thống .

Là người xử lí mọi sai sót và vấn đề khi phát sinh trong hệ thống.



* + 1. **Mô Tả các use case trong hệ thống**

### ***Đăng Nhập***

1. Tóm Tắt Định Danh

Tiêu Đề: Đăng nhập hệ thống.

Tóm Tắt: Use case này cho phép khách hàng và quản lí đăng nhập vào hệ thống.

Actor: Khách hàng và quản lí.

Ngày tạo:

Ngày cập nhật:

1. Mô Tả kịch bản

a) Điều kiện tiên quyết:

-phải đăng kí tài khoản cũng như tài khoản phải được xác thực trên hệ thống.

b) Kịch bản bình thường:

1. Hê thống kiểm tra tài khoản có hợp lệ hay không.

2. Đăng nhập thành công.

c) Các kịch bản khác:

1. Đăng nhập thất bại vì sai tài khoản hoặc mật khẩu.

2. Hệ thống yêu cầu nhập lại tài khoản và mật khẩu.

Trở lại kịch bản bình thường.

### ***Đăng kí***

1. Tóm Tắt Định Danh

Tiêu Đề: Đăng kí tài khoản.

Tóm Tắt: Use case này cho phép mọi người đăng kí tài khoản để đăng nhập hệ thống.

Actor: Khách hàng

1. Mô tả kịch bản

a) Điều kiện tiên quyết:

- Phải khai báo đầy đủ thông tin yêu cầu để có thể đăng kí tài khoản

b) Kịch bản bình thường

1.Người dùng gửi yêu cầu đăng ký tới hệ thống.

2.Hệ thống chấp nhận yêu cầu đăng ký của người dùng.

3.Người dùng nhập các thông tin cần thiết (username và password).

4.Hệ thống gửi username/password người dùng vừa gửi tới máy chủ.

5.Hệ thống chấp nhận và tài khoản được tạo

c) Kịch bản khác

1.Máy chủ sẽ kiểm tra username/password hệ thống vừa gửi. Nếu thông tin đã tồn tại, máy chủ sẽ gửi lại thông báo cho hệ thống, rồi hệ thống gửi cho người dùng yêu cầu nhập lại thông tin khác.

2. Nếu thông tin không trùng với các thông tin đang có trong dữ liệu thì máy chủ gửi cho hệ thống thông báo đăng ký thành công và hệ thống gửi lại cho người dùng.

### ***Thuê Phòng Trọ***

1. Tóm Tắt Định Danh

Tiêu Đề:Cách Thuê Phòng Trọ.

Tóm Tắt: Use case này cho phép mọi người thuê phòng trong hệ thống.

Actor: Khách hàng

1. Mô tả kịch bản

a)Điều kiện kiên quyết:

-Phải đăng nhập được vào hệ thống.

b) Kịch bản bình thường:

1. Người dùng có thể tìm kiếm phòng muốn thuê bằng ID phòng hoặc tên phòng. Hệ thống sẽ gửi ID hoặc tên về máy chủ. Máy chủ sẽ dựa vào ID hoặc tên phòng để tìm kiếm thông tin của phòng rồi gửi về hệ thống. Sau đó hệ thống gửi trả người dùng thông tin của phòng mà người dùng vừa tìm kiếm.

2. Người dùng có thể lựa chọn số lượng phòng mà người dùng muốn thuê. Hệ thống sẽ tính toán dựa vào số lượng phòng mà người dùng muốn thuê để in ra hoá đơn tiền.

3. Để có thể hoàn tất quá trình thuê phòng, người dùng phải điền thông tin cá nhân vào mẫu đã có sẵn mà hệ thống gửi. Sau đó hệ thống gửi thông tin khách hàng về máy chủ để có thể thuận tiện trong việc hỗ trợ khách hàng.

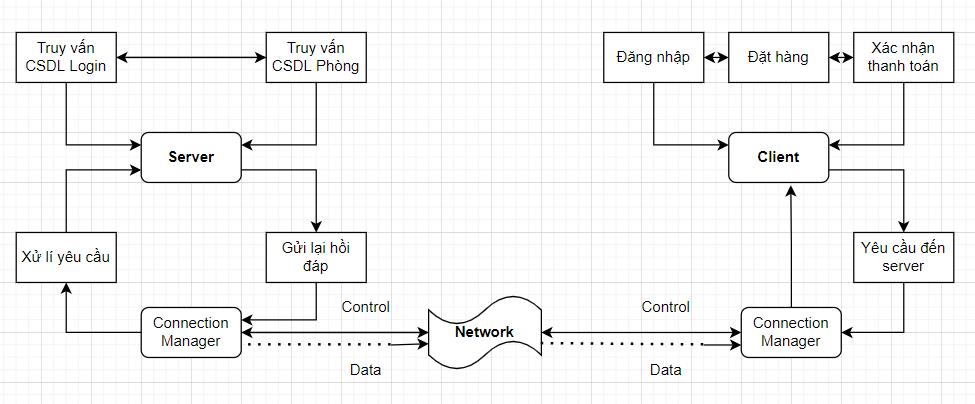
4. Hoàn tất quá trình thuê phòng.

c) Trường hợp khác.

1. Nếu ID hoặc tên phòng đã hết chỗ thì máy chủ sẽ báo lại cho hệ thống. Hệ thống sẽ thống báo hết phòng với người chơi và người chơi sẽ phải chọn lại phòng khác.

Trở lại kịch bản bình thường.

* + 1. **Kiến trúc hệ thống**



Mô hình Client server cho phép mạng tập trung các ứng dụng và chức năng tại một hoặc nhiều máu dịch vụ file chuyên dụng. Các máy này trở thành trung tâm của hệ thống. Hệ điều hành của Client server cho phép người dùng chia sẻ đồng thời cùng một tài nguyên, không quan trọng vị trí địa lý.

**Mô hình web client-server** là một mô hình nổi tiếng trong mạng máy tính, được áp dụng rất rộng rãi và là mô hình của mọi trang web hiện có. Một mô hình ngược lại là mô hình master-slaver, trong đó máy chủ (đóng vai trò ông chủ) sẽ gửi dữ liệu đến máy con (đóng vai trò nô lệ) bất kể máy con có cần hay không.

**Mô hình client/server** như sau: Client/Server là mô hình tổng quát nhất, trên thực tế thì một server có thể được nối tới nhiều server khác nhằm làm việc hiệu quả và nhanh hơn. Khi nhận được 1 yêu cầu từ client, server này có thể gửi tiếp yêu cầu vừa nhận được cho server khác ví dụ như database server vì bản thân nó không thể xử lý yêu cầu này được. Máy server có thể thi hành các nhiệm vụ đơn giản hoặc phức tạp.

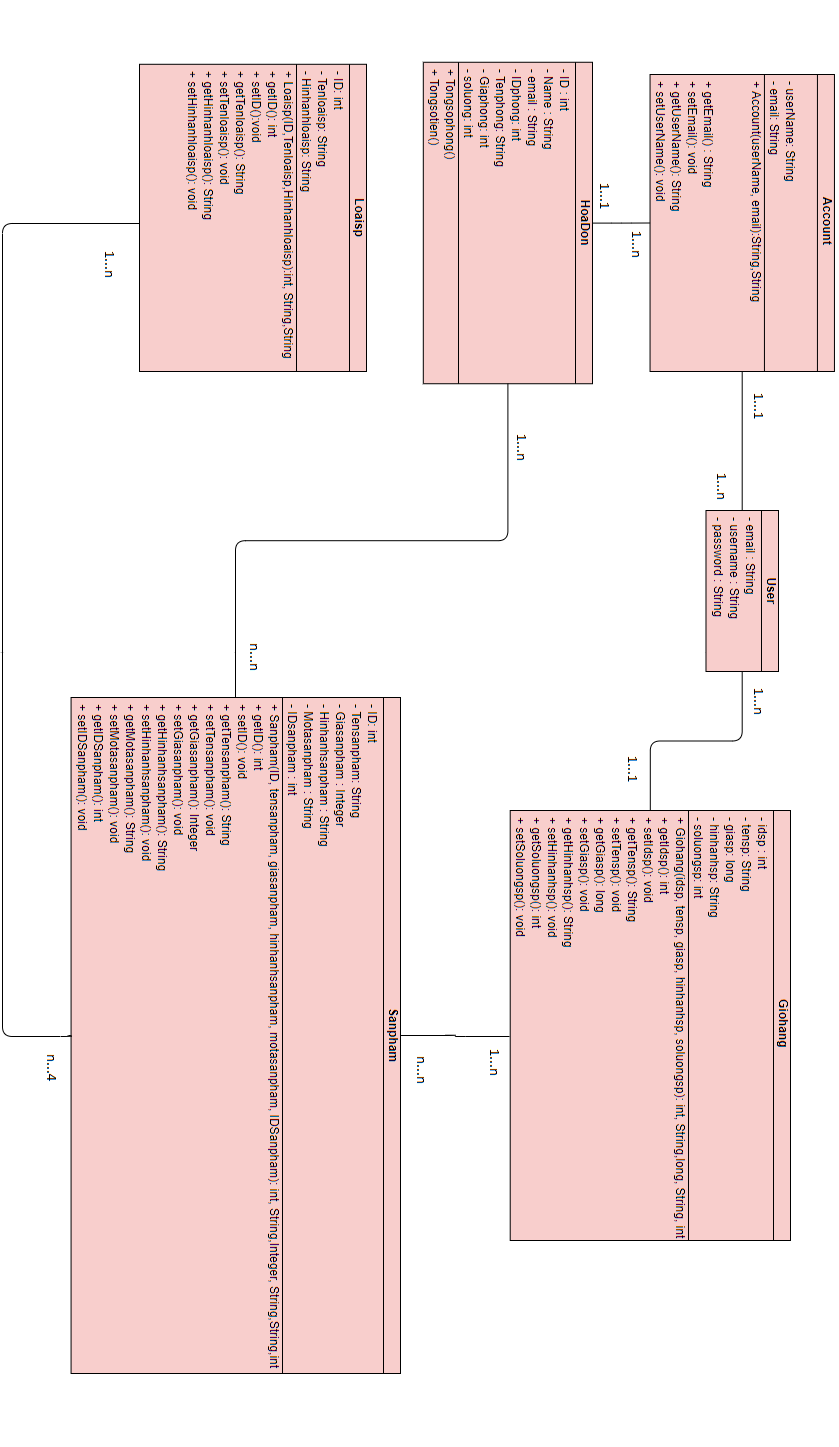
+) Client muốn đăng nhập sẽ phải gửi yêu cầu về cho server qua network và server sẽ truy vấn trong CSDL và nếu có sẽ gửi lại hồi đáp cho Connection Manager rồi gửi dữ liệu về cho client qua network.

\*NOTE: Nếu Client chưa có tài khoản đăng nhập. Client có thể tạo 1 tài khoản mới bằng cách điền thông tin và gửi yêu cầu đăng ký tài khoản tới server. Server sẽ xử lí yêu cầu rồi thêm dữ liệu vào CSDL và gửi hồi đáp cho Client qua connection manager và network.

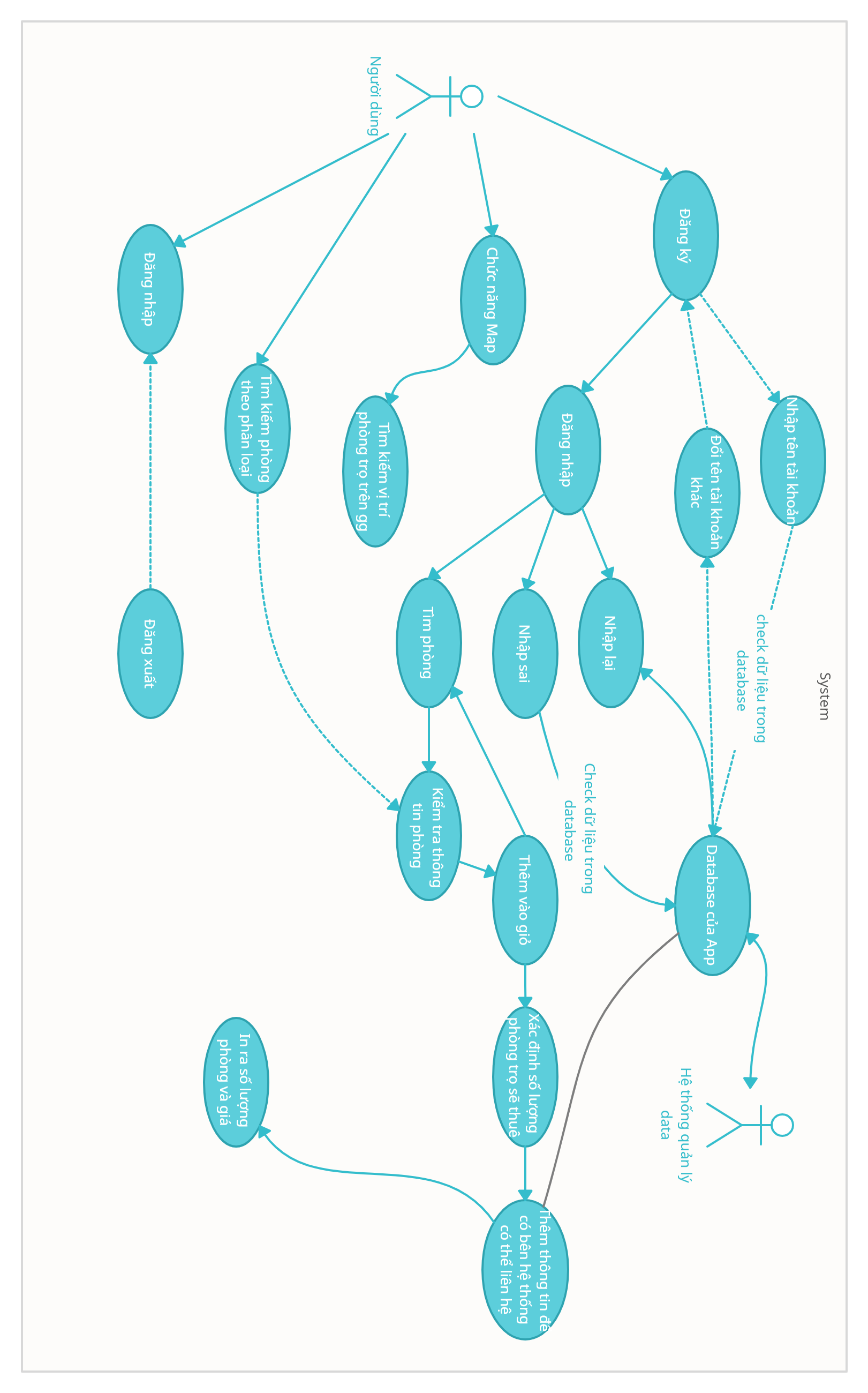
+) Sau khi đăng nhập thành công thì client có thể đặt hàng rồi gửi yêu cầu tới server qua connection manager. Server sẽ xử lí yêu cầu rồi truy vấn cơ sở dữ liệu rồi gửi dữ liệu hồi đáp cho client.

* + 1. **Biểu đồ**

1. **Biểu đồ lớp**



1. **Phân tích bài toán bằng biểu đồ USE-CASE**

****

Mô tả biểu đồ use-case uml:

1. Nhánh đăng ký :

Người dùng Nhập tên tài khoản + password. Dữ liệu được gửi đến database. Kiểm tra dữ liệu trong database, nếu như đã có dữ liệu trùng lặp thì sẽ yêu cầu người dùng thay đổi tên tài khoản.

Đăng ký xong app sẽ yêu cầu người dùng nhập lại tài khoản và password một lần nữa. Nếu như nhập sai thì sẽ yêu cầu người dùng nhập lại.

Nếu nhập đúng thì người dùng có thể vào trong giao diện của app để tìm kiếm, lựa chọn, kiểm tra thông tin phòng phù hợp với nhu cầu.

Phòng mà người dùng đã chọn sẽ được thêm vào giỏ đồ thuê phòng. Trong này người dùng có thể tuỳ chỉnh số lượng phòng muốn thuê.

Lựa chọn xong hệ thống sẽ yêu cầu người dùng nhập thông tin bản thân rồi gửi về database để có thể thuận tiện trong việc liên hệ. Rồi in ra hoá đơn cũng như thông tin số lượng phòng mà người dùng đã chọn.

1. Nhánh chức năng Map

Ở phần chức năng Map, người dùng có thể tìm kiếm các phòng trọ trên bản đồ của Hà Nội. Mỗi chấm vị trí sẽ dẫn tới địa chỉ liên kết của phòng trọ đó.

1. Nhánh tìm kiếm phòng theo phân loại

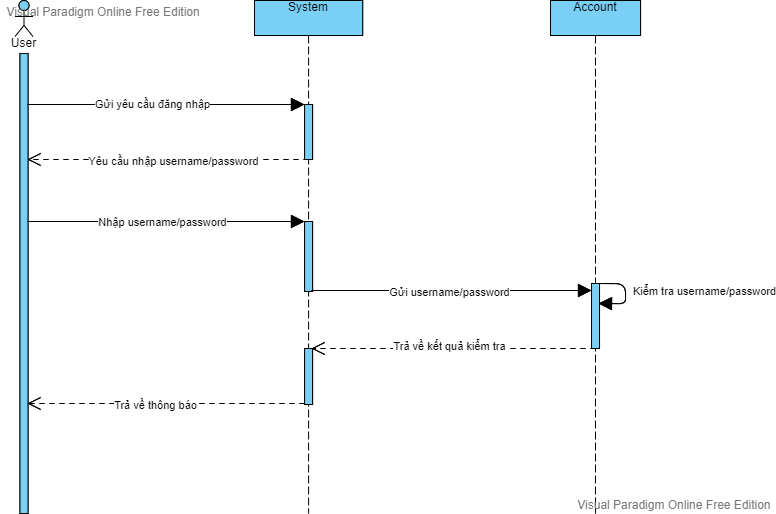
Người dùng có thể tìm kiếm phòng trọ theo kiểu đã phân loại.

1. Nhánh đăng nhập

Đơn giản là đăng nhập và đăng xuất như các app khác.

1. **Biểu đồ UML**

* **Login**



Đặc tả hành vi login

1. Người dùng gửi yêu cầu đăng nhập tới hệ thống.

2. Hệ thống chấp nhận yêu cầu đăng nhập của người dùng.

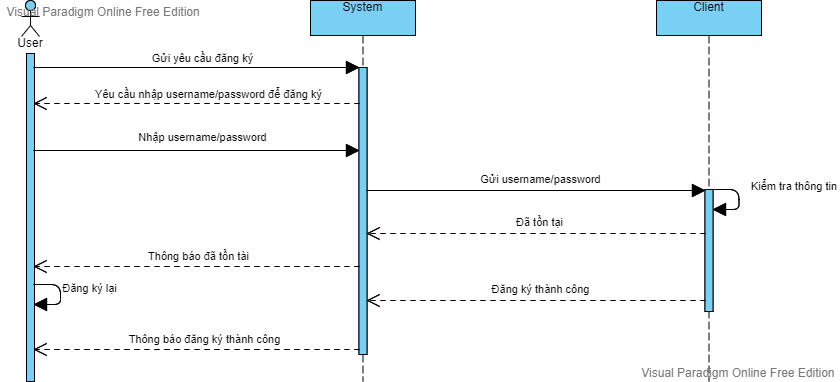
3. Người dùng nhập các thông tin username, password rồi gửi tới máy chủ

4. Thông tin được kiểm tra tại máy chủ:

- Nếu username/password chính xác thì trả về đúng.

- Nếu username/password sai thì trả về “bạn hãy nhập lại”.

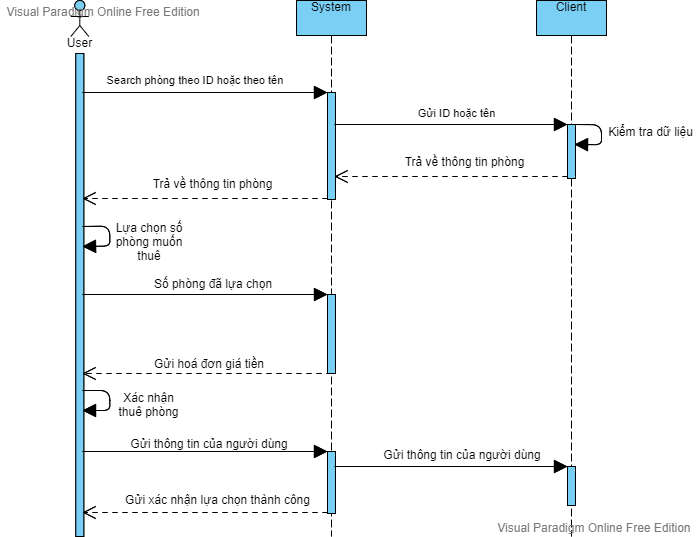
* **Sign up**



Đặc tả hành vi signup:

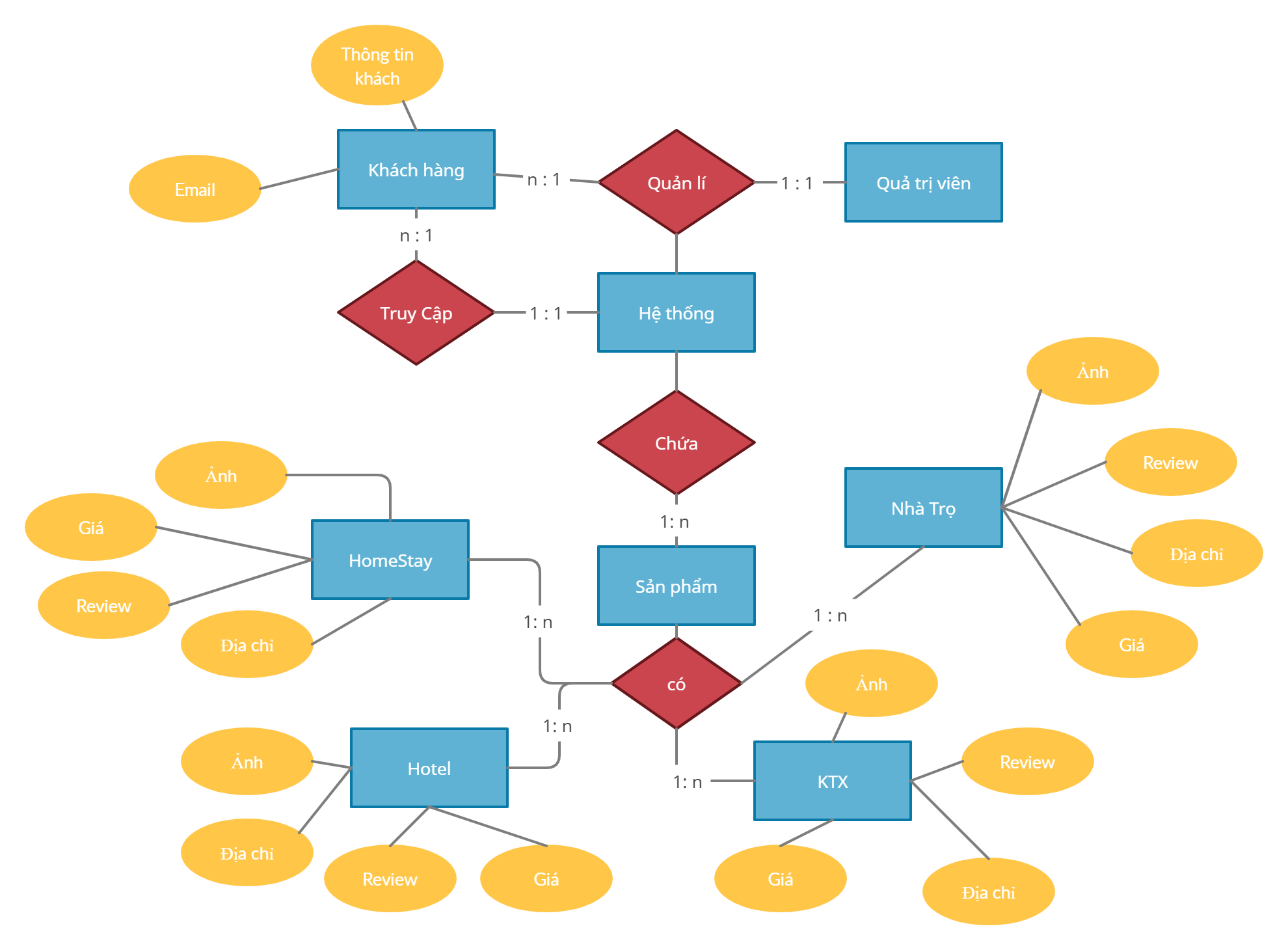
1. Người dùng gửi yêu cầu đăng ký tới hệ thống.
2. Hệ thống chấp nhận yêu cầu đăng ký của người dùng.
3. Người dùng nhập các thông tin cần thiết (username và password).
4. Hệ thống gửi username/password người dùng vừa gửi tới máy chủ.
5. Máy chủ sẽ kiểm tra username/password hệ thống vừa gửi. Nếu thông tin đã tồn tại, máy chủ sẽ gửi lại thông báo cho hệ thống, rồi hệ thống gửi cho người dùng yêu cầu nhập lại thông tin khác.
6. Nếu thông tin không trùng với các thông tin đang có trong dữ liệu thì máy chủ gửi cho hệ thống thông báo đăng ký thành công và hệ thống gửi lại cho người dùng.

* **Xác nhận thuê phòng**

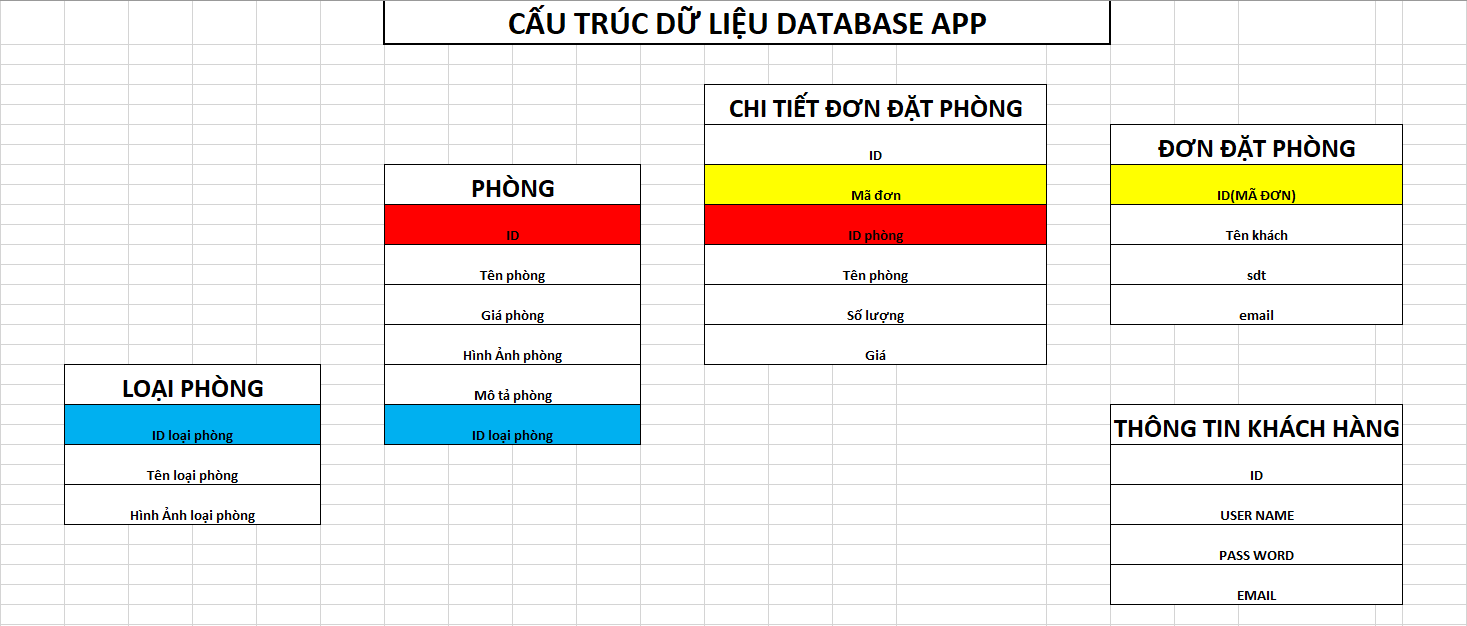


Đặc tả hành vi xác nhận thuê phòng :

1. Người dùng có thể tìm kiếm phòng muốn thuê bằng ID phòng hoặc tên phòng. Hệ thống sẽ gửi ID hoặc tên về máy chủ. Máy chủ sẽ dựa vào ID hoặc tên phòng để tìm kiếm thông tin của phòng rồi gửi về hệ thống. Sau đó hệ thống gửi trả người dùng thông tin của phòng mà người dùng vừa tìm kiếm.
2. Người dùng có thể lựa chọn số lượng phòng mà người dùng muốn thuê. Hệ thống sẽ tính toán dựa vào số lượng phòng mà người dùng muốn thuê để in ra hoá đơn tiền.
3. Để có thể hoàn tất quá trình thuê phòng, người dùng phải điền thông tin cá nhân vào mẫu đã có sẵn mà hệ thống gửi. Sau đó hệ thống gửi thông tin khách hàng về máy chủ để có thể thuận tiện trong việc hỗ trợ khách hàng.
4. Hoàn tất quá trình thuê phòng.
5. **Biểu đồ quan hệ**



* + 1. **Cấu trúc dữ liệu DATABASE**

****

* + 1. **Xây dựng các lớp**

1. **CheckConnection.java**

Kiểm tra trạng thái kết nối internet thông qua Bộ thu phát sóng:

**package** ultil;  
  
**import** android.annotation.SuppressLint;  
**import** android.content.Context;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.widget.Toast;  
  
**public class** CheckConnection {  
 **public static boolean** haveNetworkConnection(Context context){  
 **boolean** haveConnectedWifi = **false**;  
 **boolean** haveConnectedMobile = **false**;  
 ConnectivityManager cm = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY\_SERVICE);  
 @SuppressLint(**"MissingPermission"**) NetworkInfo[] netInfo = cm.getAllNetworkInfo();  
 **for** (NetworkInfo ni:netInfo){  
 **if**(ni.getTypeName().equalsIgnoreCase(**"Wifi"**))  
 **if** (ni.isConnected())  
 haveConnectedWifi=**true**;  
 **if**(ni.getTypeName().equalsIgnoreCase(**"Mobile"**))  
 **if** (ni.isConnected())  
 haveConnectedMobile=**true**;  
 }  
 **return** haveConnectedWifi || haveConnectedMobile;  
 }  
 **public static void** ShowToast\_Short(Context context,String thongbao){  
 Toast.makeText(context, thongbao, Toast.LENGTH\_SHORT).show();  
 }  
}

1. **MainActive.java**

**Activity** đại diện cho một chức năng của app, là một giao diện màn hình, nơi user tương tác trực tiếp với app. Một ứng dụng có thể có một hoặc nhiều Activity. Một Activity từ khi được gọi đến khi kết thúc sẽ có những trạng thái (state) khác nhau.

MainActivity là giao diện màn hình chính. Nơi user tương tác đầu tiên và là nơi bắt đầu của các Activity con khác.

protected void onCreate(Bundle savedInstanceState) {

...}

// Hàm kiểm tra kết nối. Nếu có kết nối thì sẽ gọi đến các hàm con và lấy dữ liệu.

public boolean onCreateOptionsMenu(Menu menu) {

...}

// Hàm lấy dữ liệu thanh menu (giỏ hàng và logout)

public boolean onOptionsItemSelected(MenuItem item) {

...}

// Hàm thực hiện lấy dữ liệu giỏ hàng và đăng xuất

protected void onStart() {

...}

// Hàm khởi chạy trương trình

private boolean authenticate() {

...}

// Hàm gọi đến phần xác thực đăng nhập

**...**

1. **Các lớp chính (USER)**

* **Loaisp.java**

**package** model;  
  
**public class** Loaisp {  
 **public int ID**;  
 **public** String **Tenloaisp**;  
 **public** String **Hinhanhloaisp**;  
  
 **public** Loaisp(**int** ID, String tenloaisp, String hinhanhloaisp) {  
 **this**.**ID** = ID;  
 **Tenloaisp** = tenloaisp;  
 **Hinhanhloaisp** = hinhanhloaisp;  
 }  
  
 **public int** getID() {  
 **return ID**;  
 }  
  
 **public void** setID(**int** ID) {  
 **this**.**ID** = ID;  
 }  
  
 **public** String getTenloaisp() {  
 **return Tenloaisp**;  
 }  
  
 **public void** setTenloaisp(String tenloaisp) {  
 **Tenloaisp** = tenloaisp;  
 }  
  
 **public** String getHinhanhloaisp() {  
 **return Hinhanhloaisp**;  
 }  
  
 **public void** setHinhanhloaisp(String hinhanhloaisp) {  
 **Hinhanhloaisp** = hinhanhloaisp;  
 }  
}

* **Sanpham.java**

**package** model;  
**import** java.io.Serializable;  
**public class** Sanpham **implements** Serializable {  
 **public int ID**;  
 **public** String **Tensanpham**;  
 **public** Integer **Giasanpham**;  
 **public** String **Hinhanhsanpham**;  
 **public** String **Motasanpham**;  
 **public int IDSanpham**;  
  
 **public** Sanpham(**int** ID, String tensanpham, Integer giasanpham, String hinhanhsanpham, String motasanpham, **int** IDSanpham) {  
 **this**.**ID** = ID;  
 **Tensanpham** = tensanpham;  
 **Giasanpham** = giasanpham;  
 **Hinhanhsanpham** = hinhanhsanpham;  
 **Motasanpham** = motasanpham;  
 **this**.**IDSanpham** = IDSanpham;  
 }  
  
 **public int** getID() {  
 **return ID**;  
 }  
  
 **public void** setID(**int** ID) {  
 **this**.**ID** = ID;  
 }  
  
 **public** String getTensanpham() {  
 **return Tensanpham**;  
 }  
  
 **public void** setTensanpham(String tensanpham) {  
 **Tensanpham** = tensanpham;  
 }  
  
 **public** Integer getGiasanpham() {  
 **return Giasanpham**;  
 }  
  
 **public void** setGiasanpham(Integer giasanpham) {  
 **Giasanpham** = giasanpham;  
 }  
  
 **public** String getHinhanhsanpham() {  
 **return Hinhanhsanpham**;  
 }  
  
 **public void** setHinhanhsanpham(String hinhanhsanpham) {  
 **Hinhanhsanpham** = hinhanhsanpham;  
 }  
 **public** String getMotasanpham() {

**return Motasanpham**;  
 }  
  
 **public void** setMotasanpham(String motasanpham) {  
 **Motasanpham** = motasanpham;  
 }  
  
 **public int** getIDSanpham() {  
 **return IDSanpham**;  
 }  
  
 **public void** setIDSanpham(**int** IDSanpham) {  
 **this**.**IDSanpham** = IDSanpham;  
 }  
}

* **Giohang.java**

**package** model;  
**public class** Giohang {  
 **public int idsp**;  
 **public** String **tensp**;  
 **public long giasp**;  
 **public** String **hinhanhsp**;  
 **public int soluongsp**;  
  
 **public** Giohang(**int** idsp, String tensp, **long** giasp, String hinhanhsp, **int** soluongsp) {  
 **this**.**idsp** = idsp;  
 **this**.**tensp** = tensp;  
 **this**.**giasp** = giasp;  
 **this**.**hinhanhsp** = hinhanhsp;  
 **this**.**soluongsp** = soluongsp;  
 }  
  
 **public int** getIdsp() {  
 **return idsp**;  
 }  
  
 **public void** setIdsp(**int** idsp) {  
 **this**.**idsp** = idsp;  
 }  
  
 **public** String getTensp() {  
 **return tensp**;  
 }  
  
 **public void** setTensp(String tensp) {  
 **this**.**tensp** = tensp;  
 }  
  
 **public long** getGiasp() {  
 **return giasp**;  
 }  
  
 **public void** setGiasp(**long** giasp) {  
 **this**.**giasp** = giasp;  
 }  
  
 **public** String getHinhanhsp() {  
 **return hinhanhsp**;  
 }  
  
 **public void** setHinhanhsp(String hinhanhsp) {  
 **this**.**hinhanhsp** = hinhanhsp;  
 }  
  
 **public int** getSoluongsp() {  
 **return soluongsp**;  
 }  
  
 **public void** setSoluongsp(**int** soluongsp) {  
 **this**.**soluongsp** = soluongsp;  
 }  
}

* **Account.java**

**package** model;  
  
**import** java.io.Serializable;  
**public class** Account **implements** Serializable {  
 **private** String **userName**;  
 **private** String **email**;  
  
 **public** Account() {  
 }  
  
 **public** Account(String userName, String email) {  
 **this**.**userName** = userName;  
 **this**.**email** = email;  
 }  
  
 **public** String getEmail() {  
 **return email**;  
 }  
  
 **public void** setEmail(String email) {  
 **this**.**email** = email;  
 }  
  
 **public** String getUserName() {  
 **return userName**;  
 }  
  
 **public void** setUserName(String userName) {  
 **this**.**userName** = userName;  
 }  
  
}

1. **Các lớp chính (Quản lý)**

* **PhongModify.java**

**Code chương trình chứa các hàm thực hiện kết nối DataBase sử dụng JDBC và các chức năng thêm, sửa, xóa, tìm.**

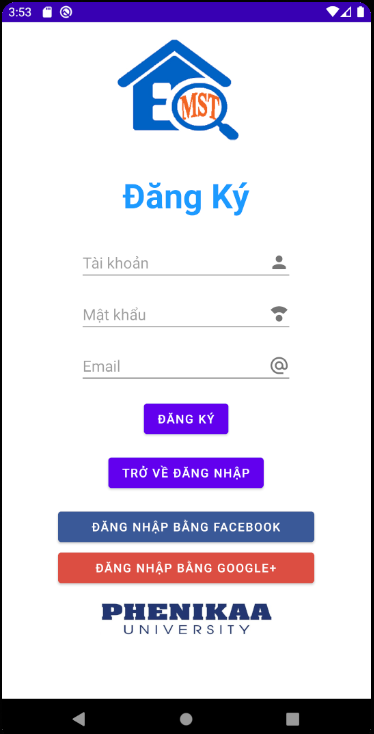
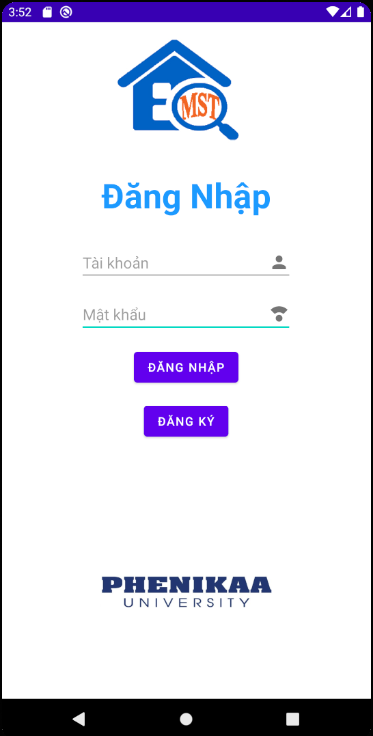
**public class** PhongModify {  
 **private static final** String ***DB\_URL*** = **"jdbc:mysql://localhost:3306/thietbi"**;  
 **private static final** String ***USER\_NAME*** = **"root"**;  
 **private static final** String ***PASSWORD*** = **""**;  
 **public static** ArrayList<Phong> findAll() **throws** SQLException{  
 ArrayList<Phong> phongList = **new** ArrayList<>();  
  
 Connection connection= **null**;  
 Statement statement =**null**;  
  
 **try**{  
 connection = DriverManager.*getConnection*(***DB\_URL***,***USER\_NAME***,***PASSWORD***);  
  
 String sql = **"select \* from sanpham"**;  
 statement = connection.createStatement();  
  
 ResultSet resultSet = statement.executeQuery(sql);  
 **while** (resultSet.next()){  
 Phong phong = **new** Phong(resultSet.getInt(**"ID"**), resultSet.getString(**"tensanpham"**),resultSet.getString(**"giasanpham"**),resultSet.getString(**"hinhanhsanpham"**), resultSet.getString(**"motasanpham"**),  
 resultSet.getString(**"IDsanpham"**));  
 phongList.add(phong);  
 }  
  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }**finally**{  
 **if**(statement!=**null**){  
 **try**{  
 statement.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
  
 **if**(connection!=**null**){  
 **try**{  
 connection.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 }  
 **return** phongList;  
 }  
 **public static void** insert(Phong phong){  
 Connection connection= **null**;  
 PreparedStatement statement =**null**;  
  
 **try**{  
 connection = DriverManager.*getConnection*(***DB\_URL***,***USER\_NAME***,***PASSWORD***);  
  
 String sql = **"insert into sanpham (ID, tensanpham, giasanpham, hinhanhsanpham, motasanpham, IDsanpham) values(?,?,?,?,?,?)"**;  
 statement = connection.prepareCall(sql);  
  
 statement.setInt(1,phong.getID());  
 statement.setString(2,phong.getTensanpham());  
 statement.setString(3,phong.getGiasanpham());  
 statement.setString(4,phong.getHinhanhsanpham());  
 statement.setString(5,phong.getMotasanpham());  
 statement.setString(6,phong.getIDsanpham());  
 statement.execute();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }**finally**{  
 **if**(statement!=**null**){  
 **try**{  
 statement.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
  
 **if**(connection!=**null**){  
 **try**{  
 connection.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 }  
 }  
 **public static void** delete(**int** id){  
 Connection connection= **null**;  
 PreparedStatement statement =**null**;  
  
 **try**{  
 connection = DriverManager.*getConnection*(***DB\_URL***,***USER\_NAME***,***PASSWORD***);  
  
 String sql = **"delete from sanpham where ID =?"**;  
 statement = connection.prepareCall(sql);  
  
 statement.setInt(1,id);  
 statement.execute();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }**finally**{  
 **if**(statement!=**null**){  
 **try**{  
 statement.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 **if**(connection!=**null**){  
 **try**{  
 connection.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 }  
  
 }**public static** ArrayList<Phong> findbyname(String name){  
 ArrayList<Phong> phongList = **new** ArrayList<>();  
 Connection connection= **null**;  
 PreparedStatement statement =**null**;  
  
 **try**{  
 connection = DriverManager.*getConnection*(***DB\_URL***,***USER\_NAME***,***PASSWORD***);  
 String sql = **"select \* from sanpham where tensanpham like ?"**;  
 statement = connection.prepareCall(sql);  
 statement.setString(1, **"%"**+name+**"%"**);  
 ResultSet resultSet = statement.executeQuery();  
 **while** (resultSet.next()){  
 Phong phong = **new** Phong(resultSet.getInt(**"ID"**), resultSet.getString(**"tensanpham"**),resultSet.getString(**"giasanpham"**),resultSet.getString(**"hinhanhsanpham"**), resultSet.getString(**"motasanpham"**),  
 resultSet.getString(**"IDsanpham"**));  
 phongList.add(phong);  
 }  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }**finally**{  
 **if**(statement!=**null**){  
 **try**{  
 statement.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 **if**(connection!=**null**){  
 **try**{  
 connection.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 }  
 **return** phongList;  
 }  
 **public static** ArrayList<Phong> findbyid(String id){  
 ArrayList<Phong> phongList = **new** ArrayList<>();  
  
 Connection connection= **null**;  
 PreparedStatement statement =**null**;  
  
 **try**{  
 connection = DriverManager.*getConnection*(***DB\_URL***,***USER\_NAME***,***PASSWORD***);  
  
 String sql = **"select \* from sanpham where ID = ?"**;  
 statement = connection.prepareCall(sql);  
 statement.setString(1, id);  
 ResultSet resultSet = statement.executeQuery();  
 **while** (resultSet.next()){  
 Phong phong = **new** Phong(resultSet.getInt(**"ID"**), resultSet.getString(**"tensanpham"**),resultSet.getString(**"giasanpham"**),resultSet.getString(**"hinhanhsanpham"**), resultSet.getString(**"motasanpham"**),  
 resultSet.getString(**"IDsanpham"**));  
 phongList.add(phong);  
 }  
  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }**finally**{  
 **if**(statement!=**null**){  
 **try**{  
 statement.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
  
 **if**(connection!=**null**){  
 **try**{  
 connection.close();  
 }**catch**(SQLException ex){  
 Logger.*getLogger*(PhongModify.**class**.getName()).log(Level.***SEVERE***,**null**,ex);  
 }  
 }  
 }  
 **return** phongList;  
 }  
}

1. **Giao diện hệ thống(USER)**

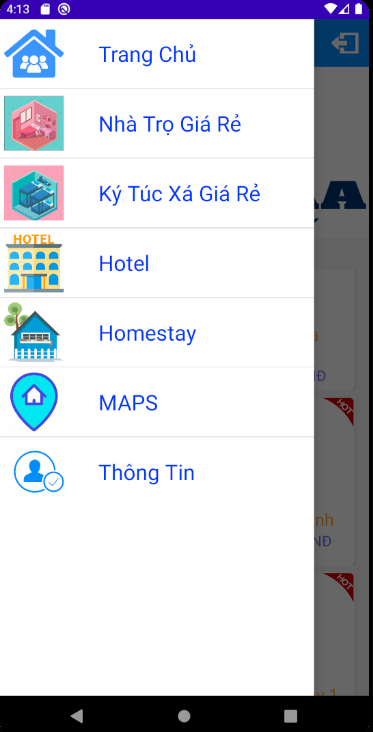
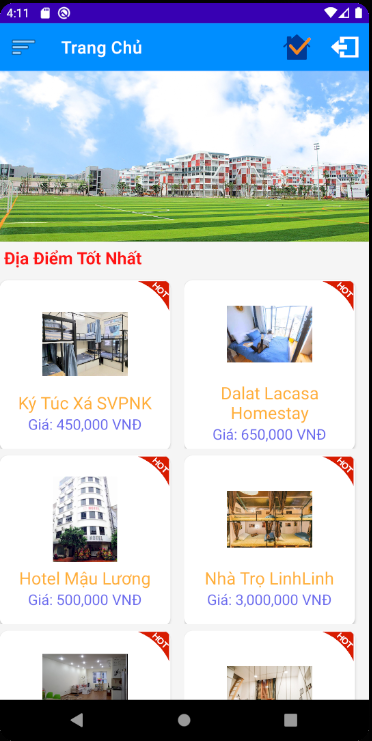
Giao diện hệ thống xây dựng theo kiểu chọn menu và điền vào mẫu cho sẵn.

Sau đây là 1 số hình ảnh của app

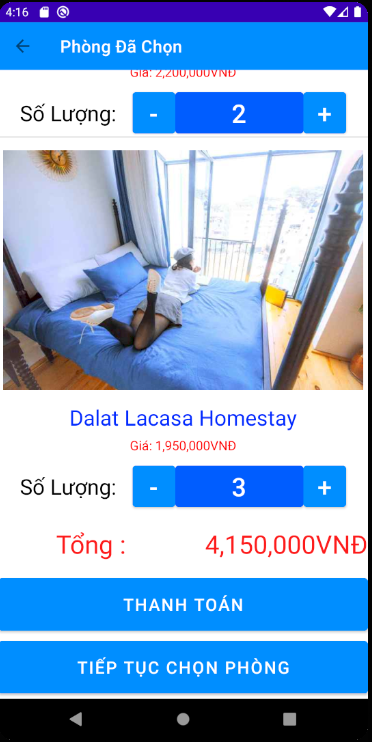
**\*\* Màn Đăng nhập | Đăng ký**



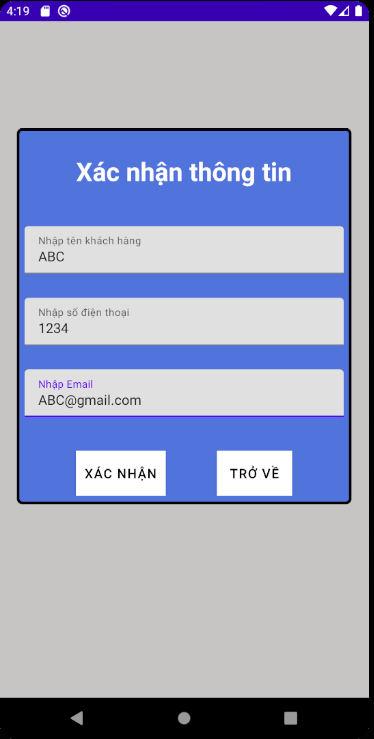
**\*\* Màn Trang chủ**



**\*\* Màn Sản phẩm và Giỏ hàng**



**\*\* Màn Xác nhận thông tin khi đồng ý Thuê phòng**



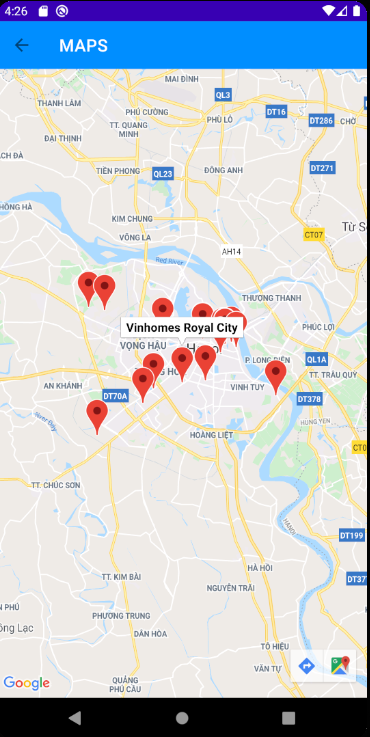
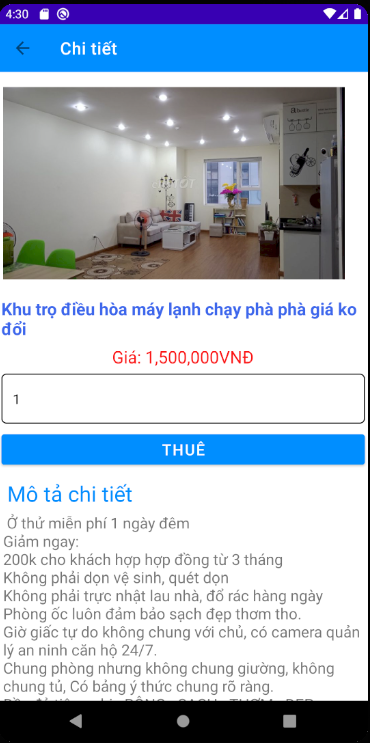
**\*\* DATABASE – bảng TT Khách hàng ()**



**\*\* DATABASE – bảng TT Đơn thuê phòng hàng của khách hàng (mã đơn hàng là mã ID của khách hàng)**



**\*\* Màn Thông tin chi tiết sản phẩm và MAP**



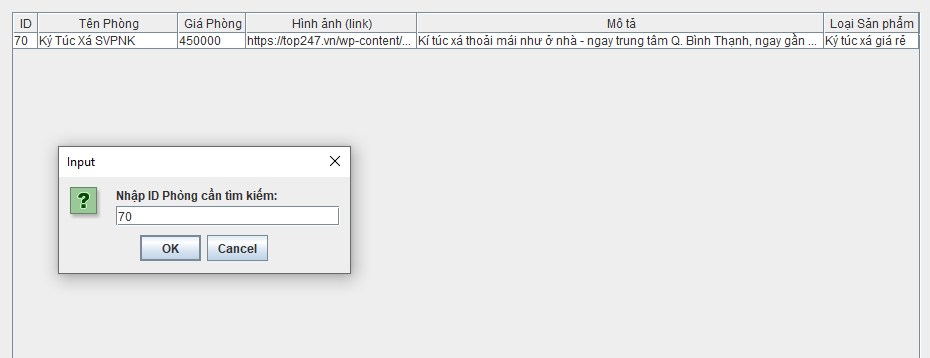
**\*\* Màn Thông tin Liên hệ Admin**



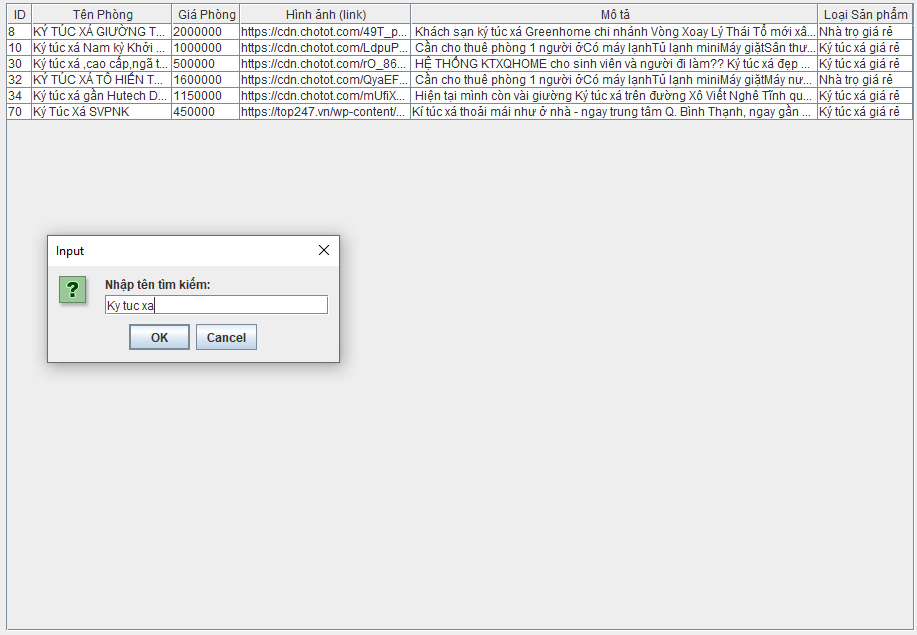
1. **Giao diện cho Chương trình (Quản lý)**

Thực hiện các chức năng quản lý hệ thống:

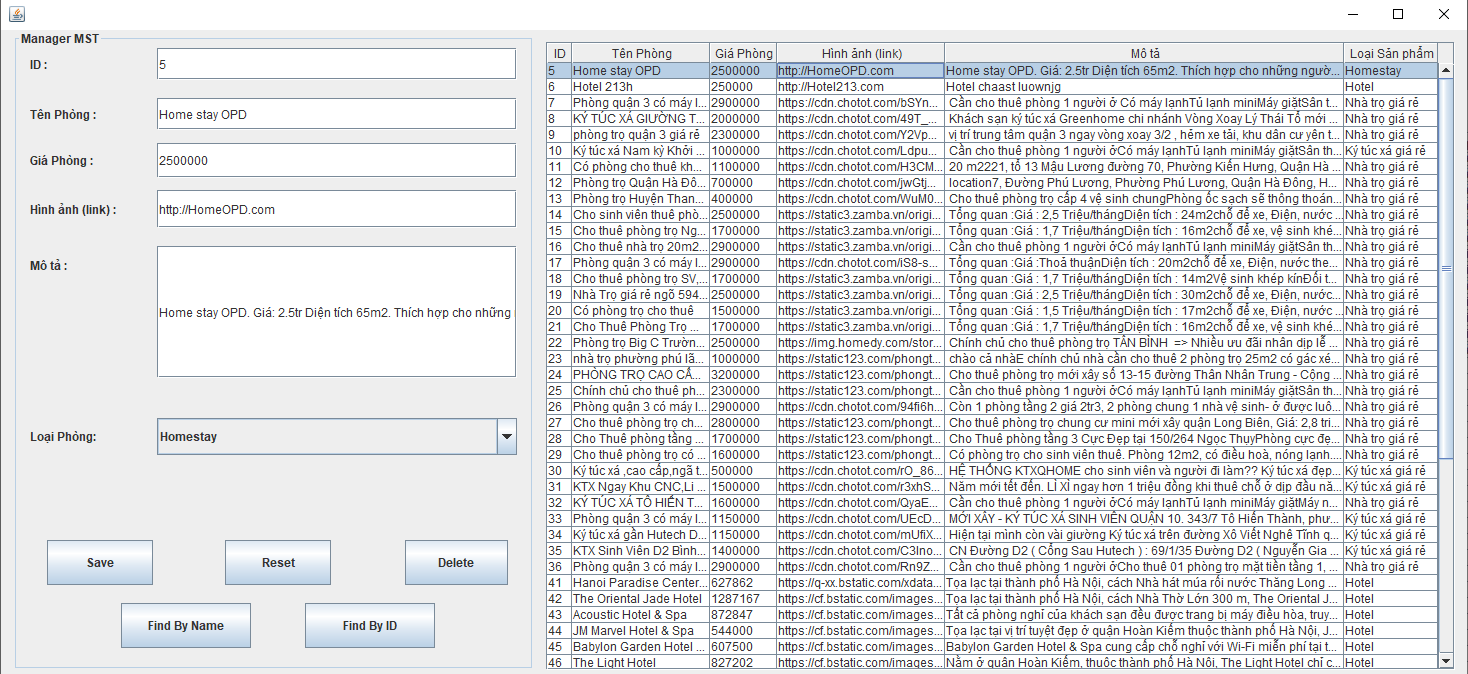
1. Tìm phòng theo ID

****

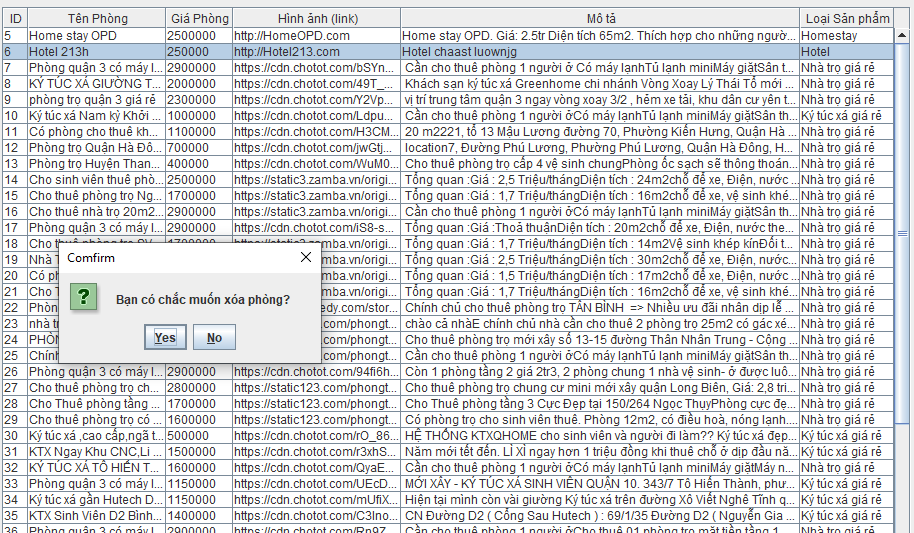
2. Tìm phòng theo Tên Phòng



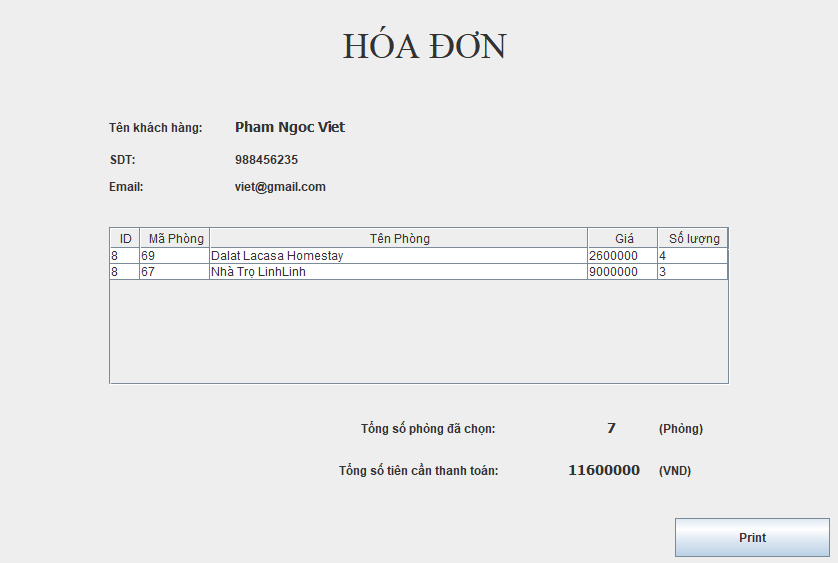
3. Thêm phòng mới



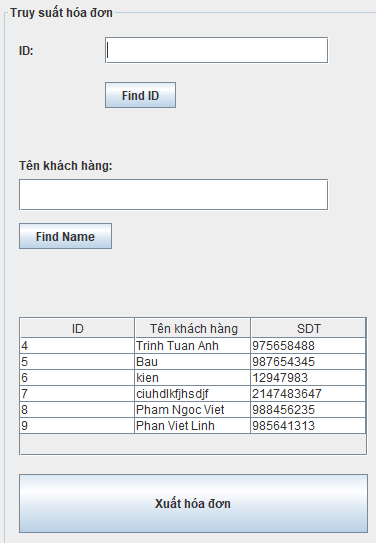
4. Xóa phòng với id



5. In Hóa Đơn



6. TT Khách hàng



1. **ỨNG DỤNG**

**\*\* Với User tải và cài đặt app tại:**

***LINK :*** https://drive.google.com/file/d/1I0DfXFGoPinrLHgseOfxSM2LYz9LkGJr/view?fbclid=IwAR0fbzK2ltrezhSKElvSoc83tPPFo5kke-Ud3ix-5t-3WclO7XQbo3PCmTQ

Đây là file APK tải về và thực hiện cài đặt

Bước 1: Vào mục tải xuống nơi bạn tải file APK về.

Bước 2: Chọn mở file APK. Tiến hành cài đặt.

Nếu không cài đặt được tiến hành các bước:

**Bước 1:** Chọn Cài đặt.

**Bước 2:** Chọn Bảo mật.

**Bước 3:** Cho phép tải các file Không rõ nguồn gốc.

**Bước 4:** Chọn file Apk.

**Bước 5:** Cho phép cài đặt.

**Bước 3: Nhấn cài đặt ứng dụng.**

**Bước 4: Tiến hành Đăng ký tài khoản.**

**Bước 5: Tiến hành Đăng nhập tài khoản vừa đăng ký.**

**Bước 6: Trải nghiệm ứng dụng đặt và thuê phòng.**

**\*\* Với Quản trị viên:**

**Thực hiện chạy Demo Chương trình Newclass.java và Newclass1.java**

1. **KẾT LUẬN**

Vậy là với mục đích ban đầu nhằm phục vụ cũng nhưng giúp đỡ mọi người có thể tìm được 1 chỗ trọ ứng ý và dễ dàng quản lý thông tin người dùng, chương trình của Nhóm bọn em đã hoàn thành như thế.

Áp dụng những tài liệu về java cũng như lập trình hướng đối tượng, app hướng đến sử thuận tiện cũng nhưng dễ dàng trong việc sử dụng cho người dùng cũng như là quản lí và để phù hợp với xu thế hiện đại hóa và online như hiện tại. Một lần nữa nhóm chúng em mong nhận được các nhận xét của thầy về đề tài này để có thể khắc phục và làm tốt hơn trong những đề tài tiếp theo.

1. **ĐÓNG GÓP TỪNG THÀNH VIÊN**

**\* Phạm Ngọc Việt:**

- Làm XML màn hình chính, menu.

- Làm menu.

- Làm localhost.

- Tạo và kết nối server.

- Viết code chương trình và kết nối app vs server.

\* **Trịnh Tuấn Anh:**

- Thu thập thông tin khách sạn, homestay.

- Tham gia hoàn thành cơ sở dữ liệu của app.

- Thiết kế giao diện hệ thống.

- Thiết kế Giao diện Hóa đơn.

- Tham gia test app trên điện thoại.

\* **Đỗ Thanh Tùng:**

- Thêm thông tin cho Database.

- Hoàn thiện các chức năng thêm, xóa.

- Hoàn thành giao diện đăng nhập đăng ký app.

- Hoàn thành đăng ký, đăng nhập tài khoản.

1. **TÀI LIỆU THAM KHẢO**

* **Java: The Complete Reference, Eleventh Edition  - By Herbert Schildt**
* **https://123docz.net//document/3841545-tai-lieu-hoc-lap-trinh-php-va-mysql-co-ban-tu-a-den-z.htm**
* **Tham khảo tạo app và cách sử dụng Android Studio và NetBeans IDE từ nhiều nguồn trên Youtube và Google. (Howkteam, KhoaPham,…)**
* [**https://stackoverflow.com/**](https://stackoverflow.com/) **(tìm và sửa lỗi)**
* [**https://online.visual-paradigm.com/**](https://online.visual-paradigm.com/) **(Vẽ các biểu đồ ER, UML, ...)**