Project: Virtual Waiter

**Mục lục**

[Chapter 1. Project description 4](#_Toc179470960)

[1.2 Problem description 4](#_Toc179470961)

[1.3 Business requirement 4](#_Toc179470962)

[1.3.1 Business rule 4](#_Toc179470963)

[1.3.2 Business analysis 4](#_Toc179470964)

[1.4 System’s actor 5](#_Toc179470965)

[1.5 Functional requirement analysis 6](#_Toc179470966)

[1.5.1 System Usecase Diagram 6](#_Toc179470967)

[1.5.2 Use case specification. 8](#_Toc179470968)

[1.6 Yêu cầu phi chức năng 9](#_Toc179470969)

[Chapter 2. Thiết kế hệ thống 10](#_Toc179470970)

[2.2 Mô hình hóa cấu trúc hệ thống 10](#_Toc179470971)

[2.2.1 Use Case Tìm kiếm 10](#_Toc179470972)

[2.2.2 Use case Xem thông tin hộ khẩu 10](#_Toc179470973)

[2.2.3 Usecase Chỉnh sửa thông tin hộ khẩu 10](#_Toc179470974)

[2.2.4 Usecase Xóa nhân khẩu 10](#_Toc179470975)

[2.2.5 Usecase Xem lịch sử thay đổi nhân khẩu 11](#_Toc179470976)

[2.2.6 Usecase Thêm nhân khẩu 11](#_Toc179470977)

[2.2.7 Usecase them hộ khẩu 11](#_Toc179470978)

[2.2.8 Usecase xem thông tin nhân khẩu 12](#_Toc179470979)

[2.2.9 Usecase thống kê 12](#_Toc179470980)

[2.2.10 Usecase Quản lý tạm trú 12](#_Toc179470981)

[2.2.11 Usecase quản lý tạm vắng 12](#_Toc179470982)

[2.2.12 Usecase Tách hộ khẩu 13](#_Toc179470983)

[2.3 Mô hình hóa sự tương tác 13](#_Toc179470984)

[2.3.1 Use Case Tìm kiếm 13](#_Toc179470985)

[2.3.2 Use case Xem thông tin hộ khẩu 13](#_Toc179470986)

[2.3.3 Usecase Chỉnh sửa thông tin hộ khẩu 14](#_Toc179470987)

[2.3.4 Usecase Xóa nhân khẩu 14](#_Toc179470988)

[2.3.5 Usecase Xem lịch sử thay đổi nhân khẩu 15](#_Toc179470989)

[2.3.6 Usecase Thêm nhân khẩu 15](#_Toc179470990)

[2.3.7 Usecase them hộ khẩu 15](#_Toc179470991)

[2.3.8 Usecase xem thông tin nhân khẩu 16](#_Toc179470992)

[2.3.9 Usecase thống kê 16](#_Toc179470993)

[2.3.10 Usecase Quản lý tạm trú 16](#_Toc179470994)

[2.3.11 Usecase quản lý tạm vắng 17](#_Toc179470995)

[2.3.12 Usecase Tách hộ khẩu 17](#_Toc179470996)

[2.4 Kiến trúc tổng thể của hệ thống 18](#_Toc179470997)

[2.4.1 Tầng trình bày 18](#_Toc179470998)

[2.4.2 Tầng ứng dụng 19](#_Toc179470999)

[2.4.3 Tầng nghiệp vụ 19](#_Toc179471000)

[2.4.4 Tầng dữ liệu 20](#_Toc179471001)

[Thiết kế chi tiết lớp 20](#_Toc179471002)

[2.4.5 Sơ đồ gói 20](#_Toc179471003)

[2.4.6 Gói Model 20](#_Toc179471004)

[2.5 Thiết kế dữ liệu 23](#_Toc179471005)

[2.5.1 Sơ đồ bảng dữ liệu 23](#_Toc179471006)

[2.6 Thiết kế giao diện 24](#_Toc179471007)

[2.6.1 Giao diện màn hình chính 24](#_Toc179471008)

[2.6.2 Chức năng thêm hộ khẩu 25](#_Toc179471009)

# Project description

## Problem description

A large restaurant needed to serve hundreds of customers at the same time. This requires processing multiple orders, servings, checkouts at the same time. It’s recommended to use an Information System to help customers to self-order, kitchen to monitor and prepare, employees to confirm orders and easily proceed checkout.

## Business requirement

### Business rule

Customer able to see and order dishes at the table through mobile devices correlated with the table.

* Dishes will have: Name, description, picture, price, size, discount.
* Discount will have 2 types: percentage discount on singular dish and combo discount. Percentage discount will automatically apply when customer select the food while combo discount will recommend customer whenever they select a dish that is in any discount combo.

Kitchen will see customer’s order and prepare the dish. Confirm that the dish is ready and waiters can serve the dish to the customer.

* Orders will have the dish ID, display name, size and number, table
* Kitchen can cancel the meal if a customer wish and they haven’t started on preparing

Waiters take the dish to serve, then confirm that the dish is served to customer.

Receptionist when check-in for customer, can check whether the tables are available and setup the tables

When check out, the system will calculate the total price and receptionist can proceed on payment.

### Business analysis

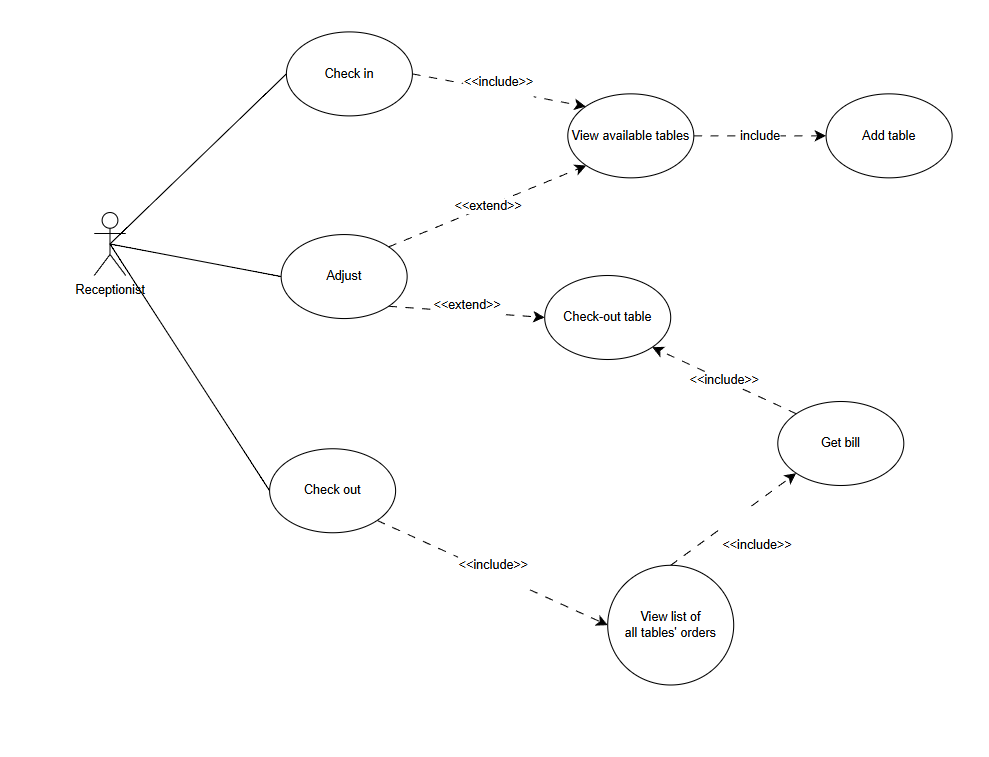
* Customer can use the mobile device to order at the table. What they can do:
  + Browse through the list of dishes
  + Select the dish, amount, size and add to cart.
  + Check the current “cart” and place order
  + Check the dishes that they have ordered and their status (preparing, served)
  + Cancel the dish that they ordered, as long as the kitchen confirm that they can.
* Kitchen can check the pending orders and set the status of the dish to preparing, ready or canceled:
  + Update the status of the order to preparing, ready, canceled.
* Waiters take completed dishes from the kitchen and serve to customers, then update the status in the order to served.
* Receptionist can organize tables, check-ins, check-outs:
  + Check whether any table available
  + Check-in for customer and setup the active table on the system
  + Change table for customer if request
  + Check-out for customer and calculate the bill.

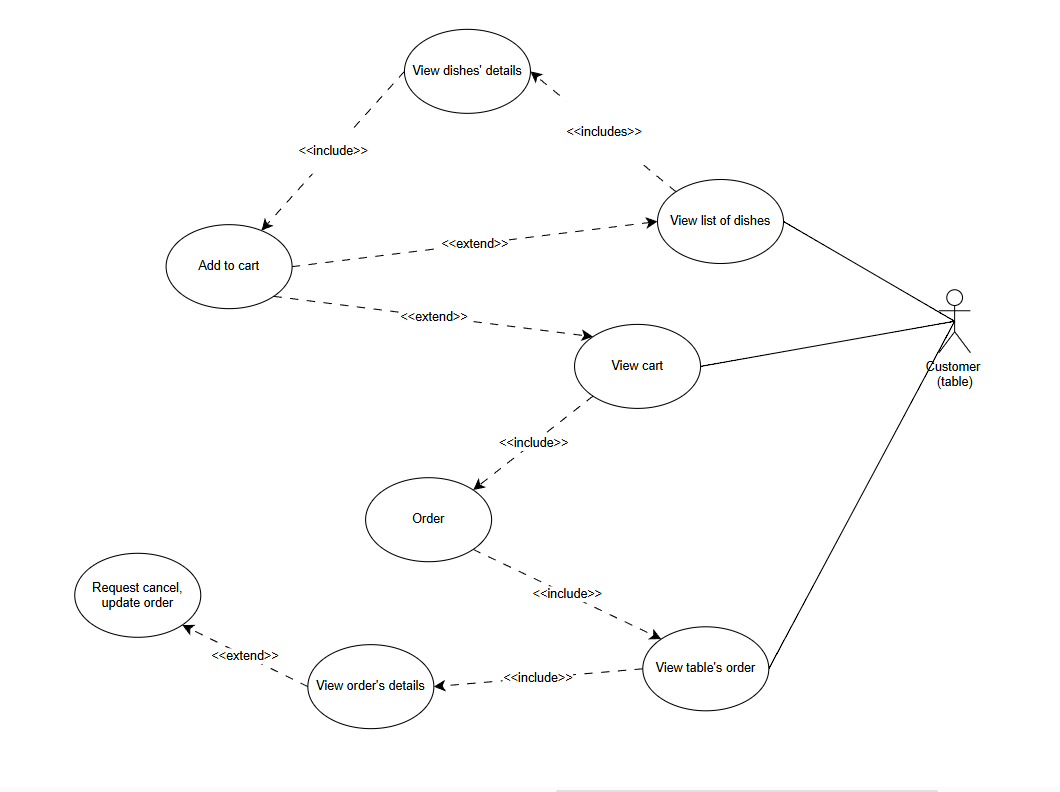
## System’s actor

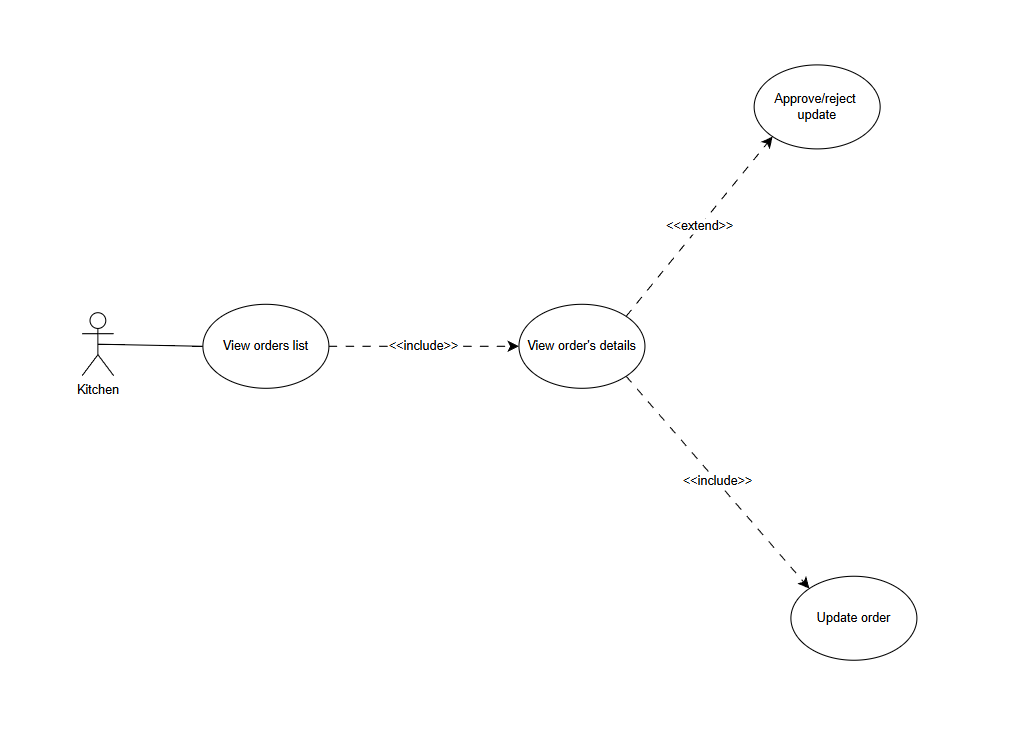
The system has 4 actors: Receptionist, Customer (mobile device on table), Kitchen, Waiter.

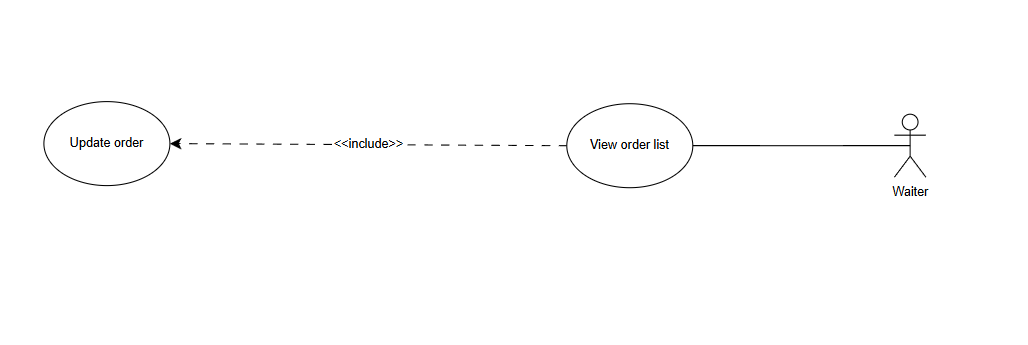
## Functional requirement analysis

### System Usecase Diagram









### Use case specification.

#### Usecase Receptionist Check-in

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC01 | **Usecase Name** | Receptionist Check-in |
| **Actor** | Receptionist | | |
| **Goals** | Check-in customer, set up tables in the name of the customer | | |
| **Triggers** | Receptionist select Check-in | | |
| **Preconditions** | None | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Receptionist | Select available tables, enter customer name | | 2 | System | Create a record for the customer | | 3 | System | Set tables as Being used | | 4 | System | Set correlation between tables and the customer record | | 5 | System | Announce succeed | | | |
| **Alternative flow** | b. Receptionist cancel. System goes back to homepage | | |
| **Post-conditions** | None | | |

#### Usecase Receptionist Adjust (remove)

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC02 | **Usecase Name** | Receptionist Adjust (remove) |
| **Actor** | Receptionist | | |
| **Goals** | Change customer’s table (remove) | | |
| **Triggers** | Receptionist select Adjust | | |
| **Preconditions** | None | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Receptionist | Select customer record | | 2 | System | Show customer’s active table | | 3 | Receptionist | Select table and remove | | 4 | System | Set table as available and refresh the cart for the tables | | 5 | System | Announce succeed, go to active table page | | | |
| **Alternative flow** | b. Receptionist cancel. System go back to homepage | | |
| **Post-conditions** | None | | |

#### Usecase Receptionist Adjust (add)

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC03 | **Usecase Name** | Receptionist Adjust (add) |
| **Actor** | Receptionist | | |
| **Goals** | Change customer’s table (add tables) | | |
| **Triggers** | Receptionist select Adjust | | |
| **Preconditions** | None | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Receptionist | Select customer record | | 2 | System | Show customer’s active table | | 3 | Receptionist | Select add | | 4 | System | Show available table | | 5 | Receptionist | Select available table(s) | | 6 | System | Set tables as Being used | | 7 | System | Set correlation between tables and the customer record | | 8 | System | Announce succeed | | | |
| **Alternative flow** | b. Receptionist cancel. System go back to homepage | | |
| **Post-conditions** | None | | |

#### Usecase Receptionist Check-out

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC04 | **Usecase Name** | Receptionist Check-out |
| **Actor** | Receptionist | | |
| **Goals** | Check-out for customer | | |
| **Triggers** | Receptionist select Check out | | |
| **Preconditions** | None | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Receptionist | Select customer record | | 2 | System | Show customer’s active table | | 3 | Receptionist | Confirms tables with customer select check out | | 4 | System | Show list of orders and calculate the bill | | 5 | Receptionist | Proceed payment and confirm | | 4 | System | Set table as available and refresh the cart for the tables | | 5 | System | Announce succeed, go to active table page | | | |
| **Alternative flow** | b. Receptionist cancel. System go back to homepage | | |
| **Post-conditions** | None | | |

#### Usecase Customer Order

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC05 | **Usecase Name** | Customer Order |
| **Actor** | Customer | | |
| **Goals** | Order dishes | | |
| **Triggers** | Customer uses the application to order | | |
| **Preconditions** | Table is active | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Customer | Select order | | 2 | System | Show list of dishes | | 3 | Customer | Select a dish | | 4 | System | Show dish details | | 5 | Customer | Select order | | 6 | System | Show order page | | 7 | Customer | Enter amount, size | | 8 | System | Calculate price (with percentage discount) | | 9 | Customer | Place order | | 10 | System | Save order, announce succeed | | 11 | Customer | Select view order list | | 12 | System | Show order list | | | |
| **Alternative flow** | 5b. Customer select combo discount  6b. System show order page of combo  7b. Customer enter amount  11c. Customer select order more  12c. System shows list of dishes | | |
| **Post-conditions** | None | | |

#### Usecase Customer Request update order

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC06 | **Usecase Name** | Customer Request update order |
| **Actor** | Customer | | |
| **Goals** | Customer request to update order | | |
| **Triggers** | Customer select update for an order | | |
| **Preconditions** | Order is not served or ready. | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Customer | Select request update | | 2 | System | Show order details edit page | | 3 | Customer | Update details and submit | | 4 | System | Receive request | | | |
| **Alternative flow** | b. Customer cancel. System go back to order’s details | | |
| **Post-conditions** | None | | |

#### Usecase Kitchen accept/reject customer update

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC07 | **Usecase Name** | Kitchen process update |
| **Actor** | Kitchen | | |
| **Goals** | Accept customer’s update or reject it | | |
| **Triggers** | Customer sent request, kitchen view will refresh to see the request | | |
| **Preconditions** | There are update from customer for the order | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | System | Shows order with updates | | 2 | Kitchen | Open order to check request | | 3 | System | Show request, comparing the changes | | 4 | Kitchen | Accept update | | 5 | System | Change order | | | |
| **Alternative flow** | 4b. Kitchen reject. System update request customer to rejected | | |
| **Post-conditions** | None | | |

#### Usecase Kitchen Update order

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC08 | **Usecase Name** | Kitchen update order |
| **Actor** | Kitchen | | |
| **Goals** | Set the order to ready so waiters can take it to serve | | |
| **Triggers** | Kitchen finish preparing the dish | | |
| **Preconditions** | There are customer order | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Kitchen | Open order detail | | 2 | System | Shows order detail | | 3 | Kitchen | Update order to Ready | | 4 | System | Announcing succeed and go back to order list | | | |
| **Alternative flow** | 3b. Kitchen cancel. System go back to order list | | |
| **Post-conditions** | None | | |

#### Usecase Waiter Update order

|  |  |  |  |
| --- | --- | --- | --- |
| **Usecase code** | UC09 | **Usecase Name** | Waiter Update order |
| **Actor** | Waiter | | |
| **Goals** | For waiter to update the order to served after serving the dish to customer | | |
| **Triggers** | Waiter successfully serve the customer | | |
| **Preconditions** | There are Ready order | | |
| **Basic flow**  **(Success)** | |  |  |  | | --- | --- | --- | | **No.** | **Proceed by** | **Action** | | 1 | Waiter | Open order details | | 2 | System | Show order details | | 3 | Waiter | Update the order to Served | | 4 | System | Update the information. And announce success | | | |
| **Alternative flow** | None | | |
| **Post-conditions** | None | | |

## Non-functional requirement

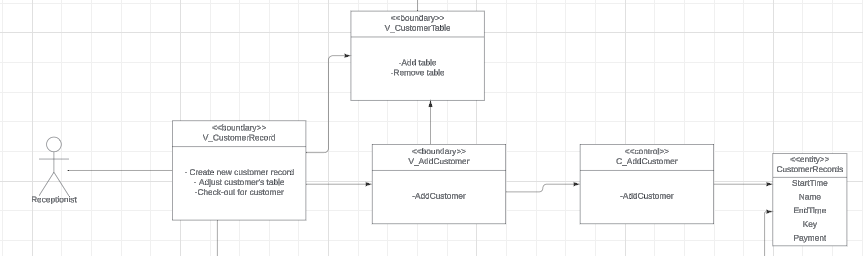
There is no non-functional requirement

# System Designing

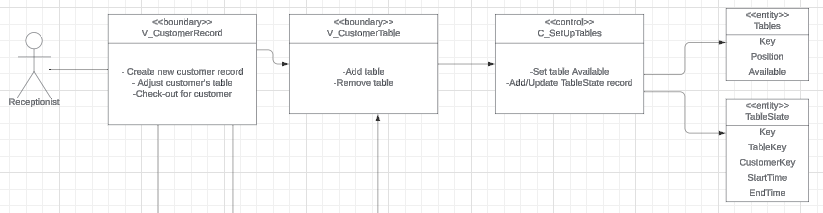
## System Structure Modeling

### Receptionist’s Usecase:

Usecase Receptionist Check-in:



Usecase Receptionist Adjust:



Usecase Receptionist Check-out

### Usecase Receptionist Adjust (remove)

### Usecase Receptionist Adjust (add)

### Usecase Receptionist Check-out

### Usecase Customer Order

### Usecase Customer Request update order

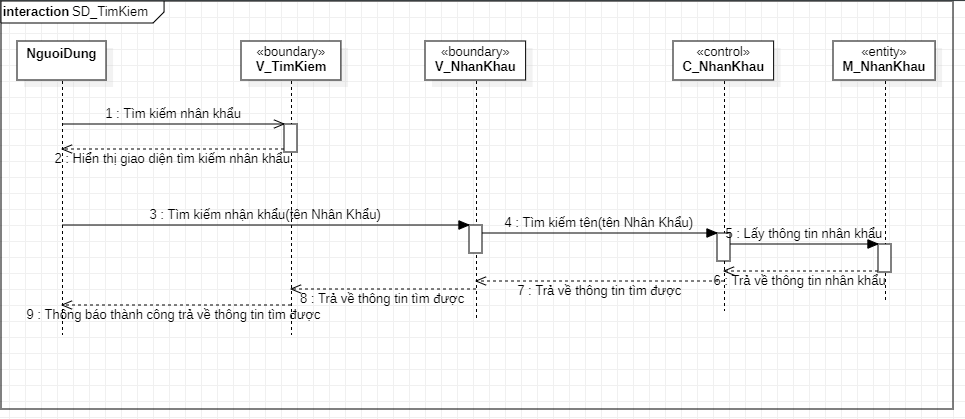
### Usecase Kitchen accept/reject customer update

### Usecase Kitchen Update order

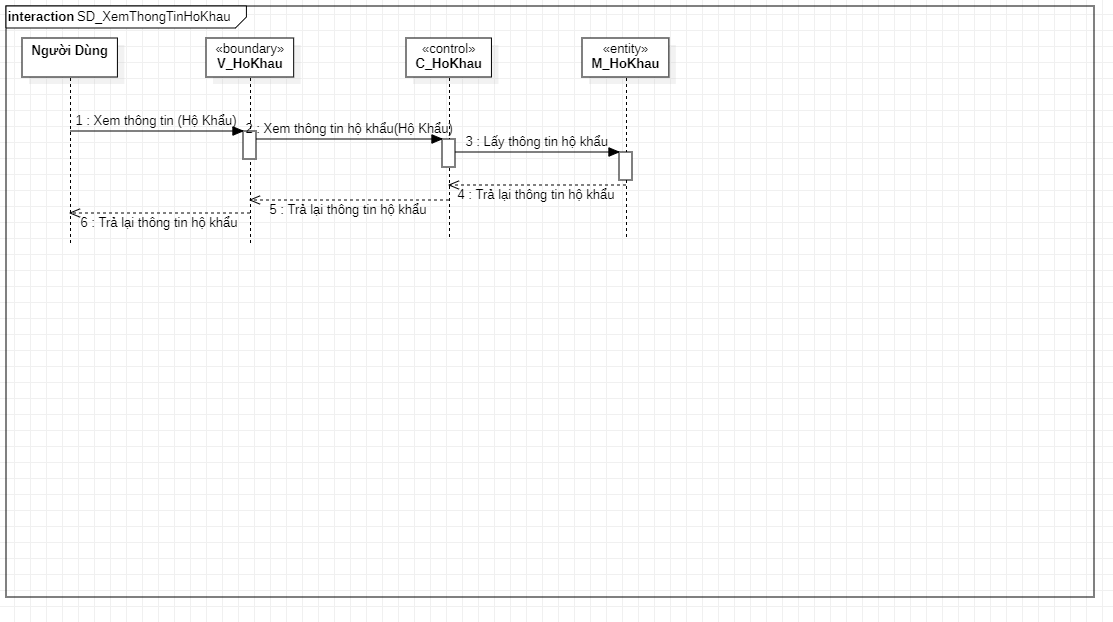
### Usecase Waiter Update order

## Mô hình hóa sự tương tác

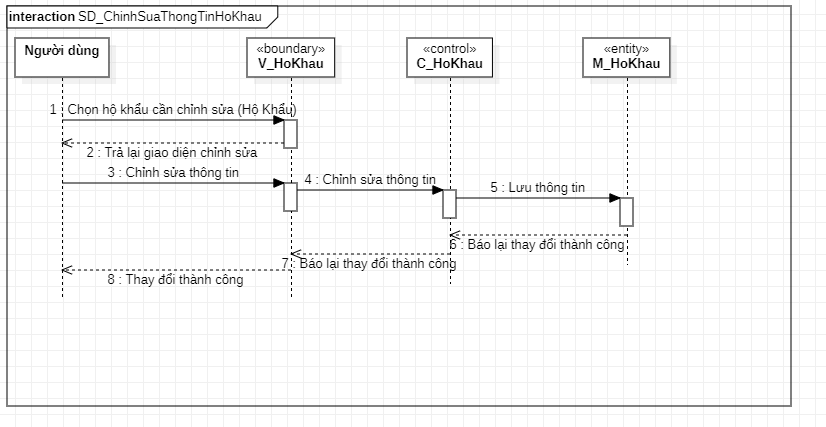
### Use Case Tìm kiếm



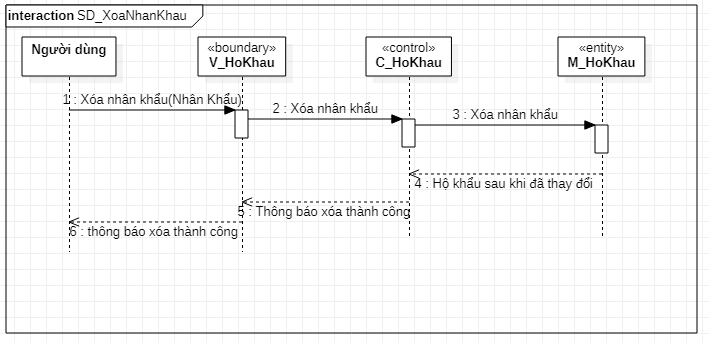
### Use case Xem thông tin hộ khẩu



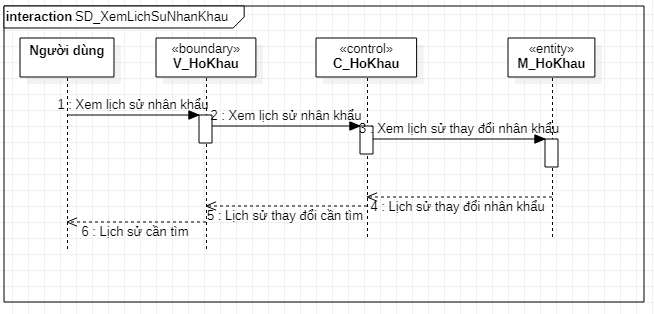
### Usecase Chỉnh sửa thông tin hộ khẩu



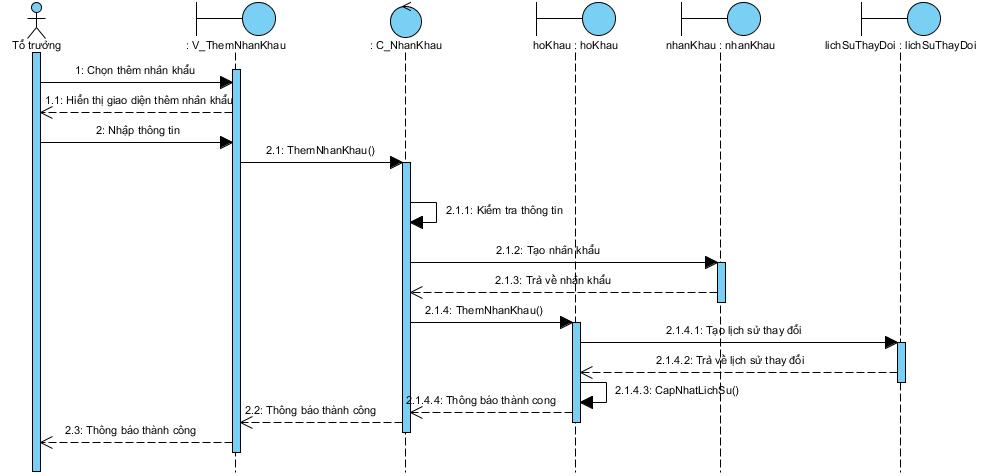
### Usecase Xóa nhân khẩu



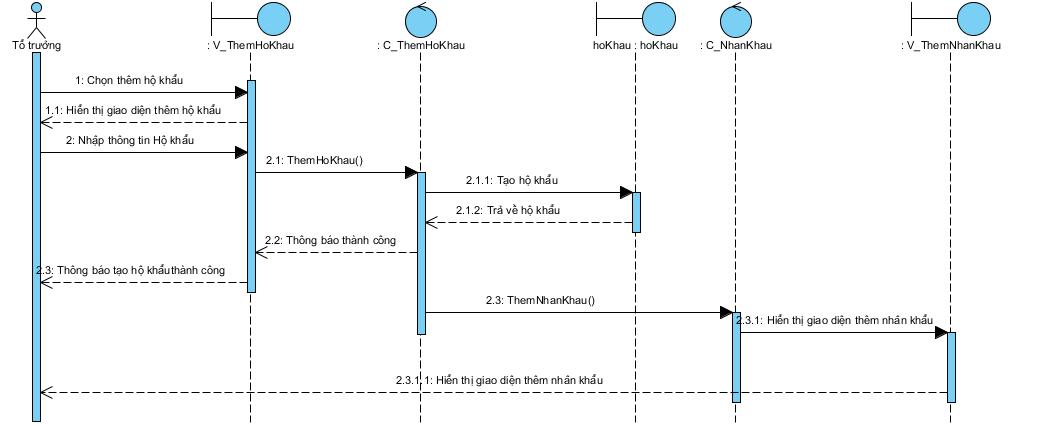
### Usecase Xem lịch sử thay đổi nhân khẩu



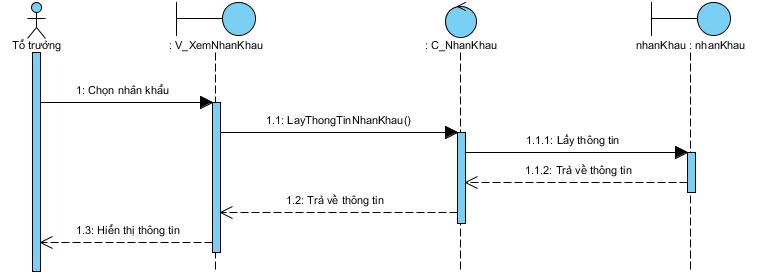
### Usecase Thêm nhân khẩu



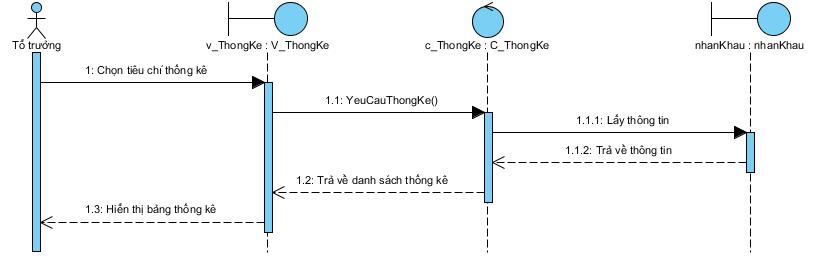
### Usecase them hộ khẩu



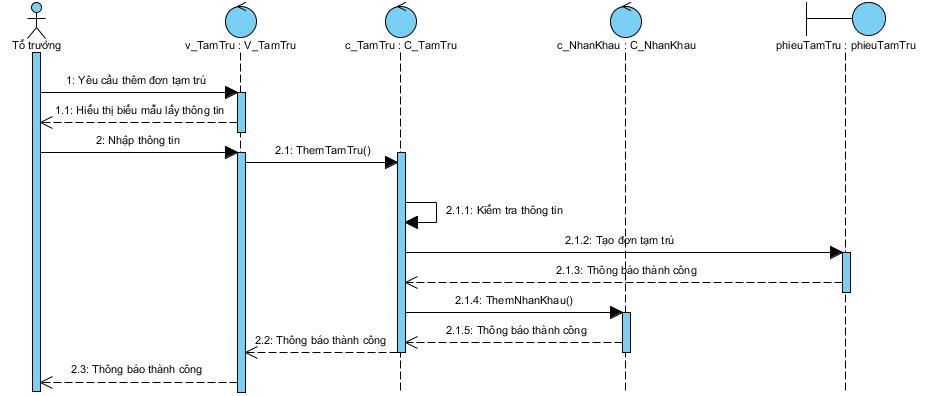
### Usecase xem thông tin nhân khẩu



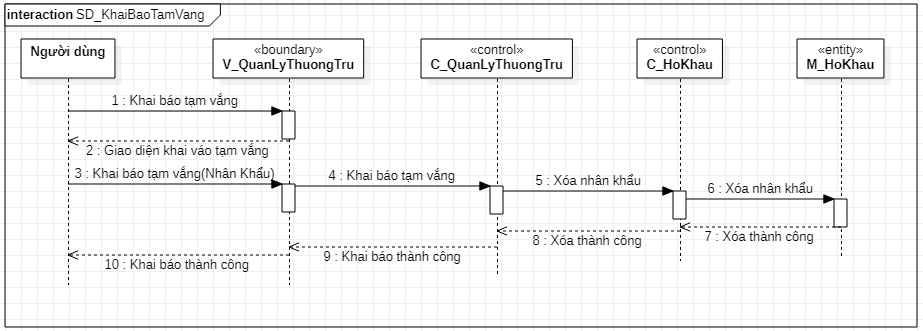
### Usecase thống kê



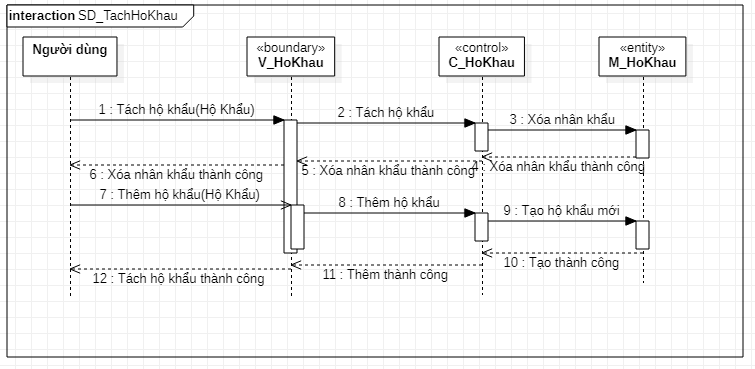
### Usecase Quản lý tạm trú



### Usecase quản lý tạm vắng

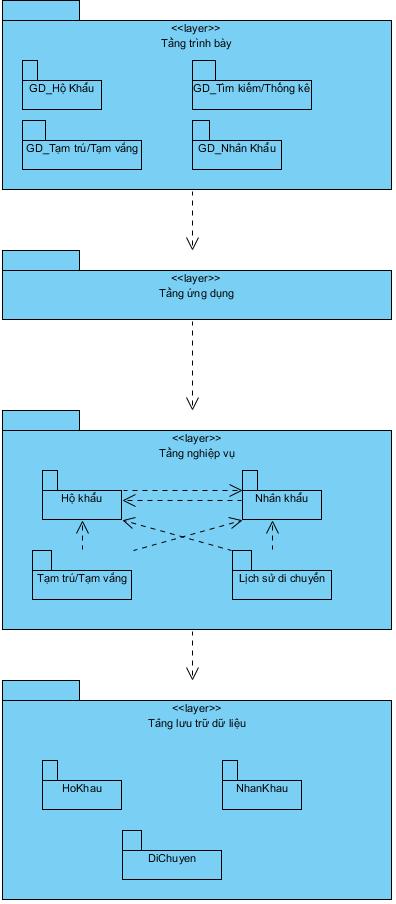


### Usecase Tách hộ khẩu



## Kiến trúc tổng thể của hệ thống

Kiến trúc phân tầng của hệ thống được thể hiện trong hình vẽ sau đây:



### Tầng trình bày

Gồm các lát:

* GD\_Hộ khẩu:
  + Thêm hộ khẩu
  + Tách hộ khẩu
  + Xem thông tin hộ khẩu
  + Chỉnh sửa thông tin hộ khẩu
  + Thêm nhân khẩu
  + Xóa nhân khẩu
  + Xem lịch sử thay đổi hộ khẩu
* GD\_Tìm kiếm/Thống kê:
  + Tìm kiếm
  + Thống kê
* GD\_Tạm trú/Tạm vắng:
  + Thêm đơn đk tạm vắng
  + Thêm đơn đk tạm trú
* GD\_Nhân khẩu:
  + Xem thông tin nhân khẩu
  + Chỉnh sửa thông tin nhân khẩu

### Tầng ứng dụng

Gồm các lát:

* ĐK-Hộ khẩu
* ĐK-Nhân khẩu
* ĐK-Tạm trú/Tạm vắng
* ĐK-Tìm kiếm/Thống kê

### Tầng nghiệp vụ

Gồm các lát:

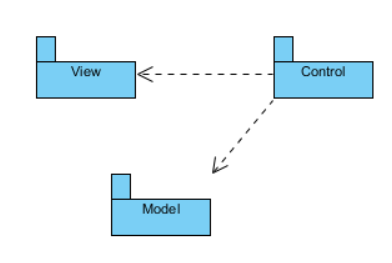
* Hộ khẩu:
  + Xem thông tin hộ khẩu
  + Chỉnh sửa thông tin hộ khẩu
  + Thêm nhân khẩu
  + Xóa nhân khẩu
  + Xem lịch sử thay đổi hộ khẩu
* Nhân khẩu:
  + Xem thông tin nhân khẩu
  + Chỉnh sửa thông tin nhân khẩu
* Lịch sử di chuyển
* Tạm trú/Tạm vắng:
  + Thêm đơn đk tạm vắng
  + Thêm đơn đk tạm trú

### Tầng dữ liệu

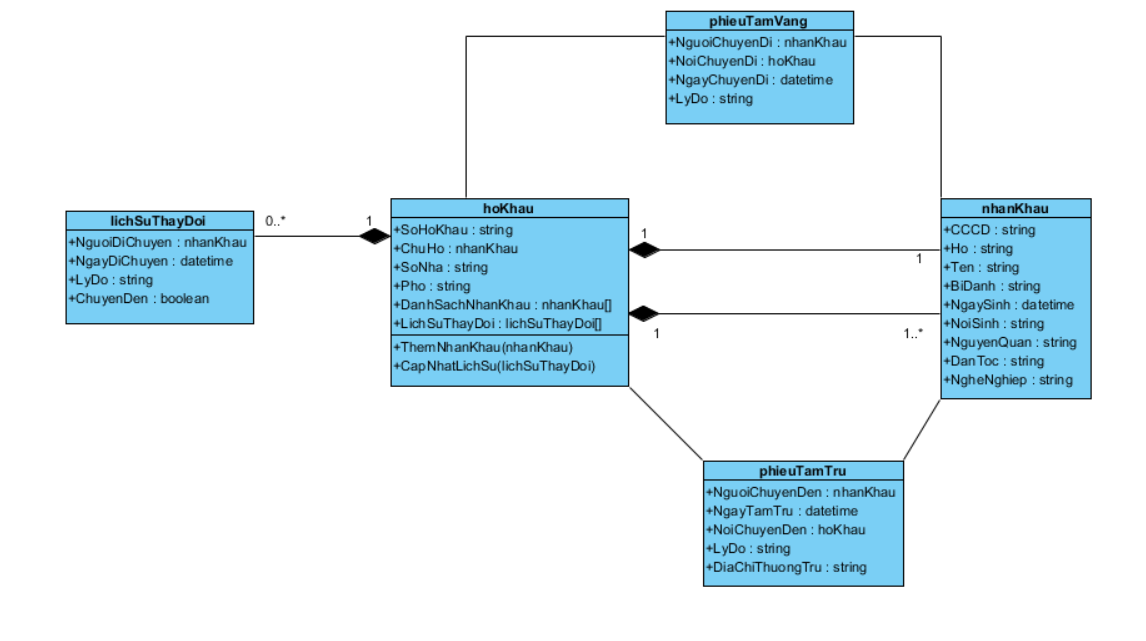
* Hộ khẩu
* Nhân khẩu
* Lịch sử di chuyển

## Thiết kế chi tiết lớp

### Sơ đồ gói



### Gói Model



#### Lớp hoKhau

* Mô tả: lớp chứa các thông tin của một hộ khẩu
* Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Phạm vi truy cập | Mô tả |
| soHoKhau | String | public | Số hộ khẩu |
| chuHo | NhanKhau | public | Nhân khẩu là chủ hộ |
| soNha | String | public | Địa chỉ số nhà |
| Pho | String | public | Địa chỉ phố |
| danhSachNhanKhau | NhanKhau[] | public | Danh sách nhân khẩu trong hộ khẩu |
| lichSuThayDoi | LichSuThayDoi[] | public | Lịch sử thay đổi nhân khẩu |

* Các phương thức

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên phương thức | Đầu vào | Đầu ra | Phạm vi truy cập | Mô tả |
| themNhanKhau | NhanKhau | void | public | Thêm một nhân khẩu vào danh sách nhân khẩu |
| capNhatLichSu | LichSuThayDoi | void | public | Cập nhật lại lịch sử thay đổi |
| Xóa nhân khẩu | NhanKhau | void | public | Xóa một nhân khẩu khỏi danh sách nhân khẩu |

#### Lớp nhanKhau

* Mô tả: lớp biểu thị thông tin của 1 nhân khẩu
* Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Phạm vi truy cập | Mô tả |
| CCCD | String | public | Căn cước công dân |
| ho | String | public | Họ của nhân khẩu |
| ten | String | public | Tên của nhân khẩu |
| biDanh | String | public | Bí danh của nhân khẩu |
| ngaySinh | DateTime | public | Ngày sinh của nhân khẩu |
| noiSinh | String | public | Nơi sinh của nhân khẩu |
| nguyenQuan | String | public | Nguyên quán của nhân khẩu |
| danToc | String | public | Dân tộc |
| ngheNghiep | String | public | Nghề nghiệp hiện tại |

#### Lớp phieuTamTru

* Mô tả: lớp biểu thị thông tin của 1 phiếu tạm trú
* Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Phạm vi truy cập | Mô tả |
| nguoiChuyenDen | NhanKhau | public | Người chuyển đến |
| ngayTamTru | DateTime | private | Ngày bắt đầu tạm trú |
| noiChuyenDen | HoKhau | private | Nơi người chuyển đến tạm trú |
| lyDo | String | private | Lý do tạm trú |
| diaChiThuongTru | String | private | Địa chỉ thường trú của người chuyển đến |

#### Lớp phieuTamVang

* Mô tả: chứa thông tin của phiếu tạm vắng khi 1 người chuyển đi
* Các thuộc tính:

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Phạm vi truy cập | Mô tả |
| nguoiChuyenDi | NhanKhau | public | Người đã chuyển đi |
| ngayChuyenDi | DateTime | public | thời gian chuyển đi |
| lyDo | String | public | Lý do chuyển đi |
| noiChuyenDi | HoKhau | public | Hộ khẩu chuyển đi |

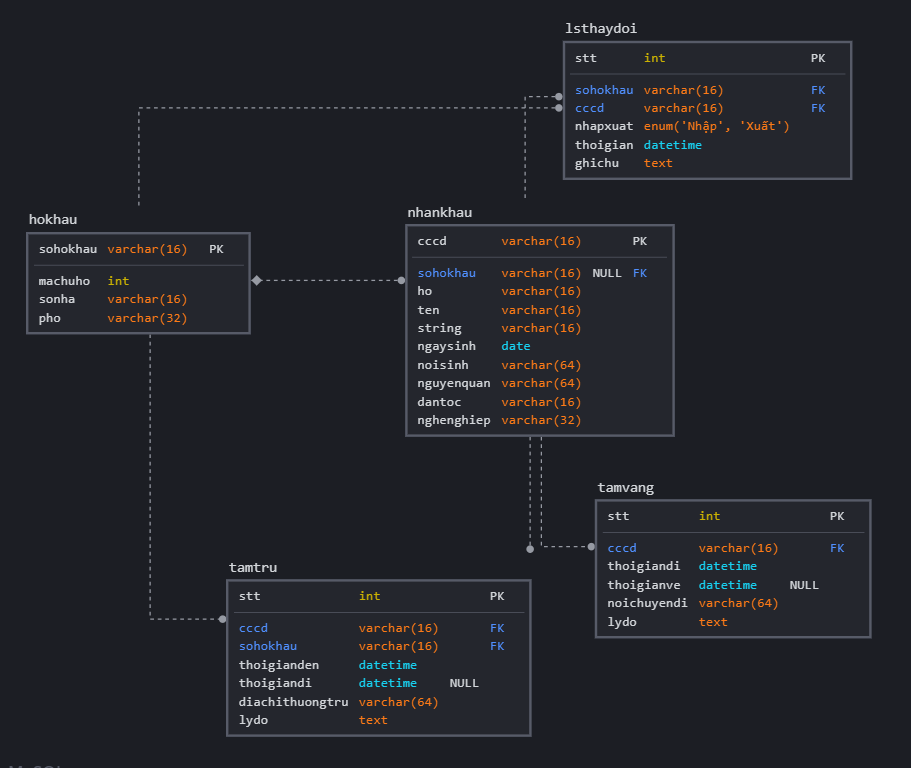
#### Lớp lichSuThayDoi

* Mô tả: lớp chứa thông tin lịch sử thay đổi của hộ khẩu
* Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| Tên thuộc tính | Kiểu dữ liệu | Phạm vi truy cập | Mô tả |
| nguoiDiChuyen | NhanKhau | public | Người đã chuyển đi hoặc đến |
| ngayDiChuyen | DateTime | public | thời gian thay đổi |
| lyDo | String | public | Lý do thay đổi hộ khẩu |
| chuyenDen | Boolean | public | Đánh dấu việc chuyển đến hoặc chuyển đi |

## Thiết kế dữ liệu

### Sơ đồ bảng dữ liệu



## Thiết kế giao diện

### Giao diện màn hình chính

### Chức năng thêm hộ khẩu