|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**  **KHOA CÔNG NGHỆ THÔNG TIN**  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  **ĐỒ ÁN MÔN HỌC**  **TÌM HIỂU VỀ ANGULAR ĐỂ XÂY DỰNG**  **HỆ THỐNG WEBSITE QUẢN LÝ KÝ TÚC XÁ**  **SINH VIÊN CHO TRƯỜNG ĐẠI HỌC NTT**   |  |  | | --- | --- | | **Giảng viên hướng dẫn :** | **Giang Hào Côn** | | **Sinh viên thực hiện :** | **Phạm Tuấn Kiệt** | | **MSSV :** | **1711543175** | | **Chuyên ngành**  **:** | **Kỹ thuật phần mềm** | | **Khóa :** | **2017** |     **Tp.HCM, tháng 01 năm 2021** |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH**  **KHOA CÔNG NGHỆ THÔNG TIN**  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  **ĐỒ ÁN MÔN HỌC**  **TÌM HIỂU VỀ ANGULAR ĐỂ XÂY DỰNG**  **HỆ THỐNG WEBSITE QUẢN LÝ KÝ TÚC XÁ**  **SINH VIÊN CHO TRƯỜNG ĐẠI HỌC NTT**   |  |  | | --- | --- | | **Giảng viên hướng dẫn :** | **Giang Hào Côn** | | **Sinh viên thực hiện :** | **Phạm Tuấn Kiệt** | | **MSSV :** | **1711543175** | | **Chuyên ngành**  **:** | **Kỹ thuật phần mềm** | | **Khóa :** | **2017** |     **Tp.HCM, tháng 01 năm 2021** |

|  |  |
| --- | --- |
| Trường Đại học Nguyễn Tất Thành  **Khoa Công Nghệ Thông Tin**  🙜 🙜 🙝 🙝 | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập – Tự do – Hạnh phúc**  🙜 🙜 🙝 🙝 |
|  |  |

**NHIỆM VỤ ĐỒ ÁN MÔN HỌC**

*(Sinh viên phải đóng tờ này vào báo cáo)*

Họ và tên: **PHẠM TUẤN KIỆT** MSSV:**1711543175**

Chuyên ngành: **Kỹ thuật phần mềm** Lớp: **17DTH1A**

Tên đề tài: **TÌM HIỂU VỀ ANGULAR ĐỂ XÂY DỰNG HỆ THỐNG WEBSITE QUẢN LÝ KÝ TÚC XÁ SINH VIÊN CHO TRƯỜNG ĐẠI HỌC NTT**

Giảng viên giảng dạy: **ThS.** **GIANG HÀO CÔN**

Thời gian thực hiện: **01/12/2021** **đến 02/01/2022**

Nhiệm vụ/nội dung (mô tả chi tiết nội dung, yêu cầu, phương pháp):

* Xác định yêu cầu, thu thập thông tin và các dữ liệu liên quan để xây dựng phần mềm.
* Xây dựng cơ sở dữ liệu trên MySql.
* Phân tích thiết kế và phác thảo giao diện các màn hình xử lý.
* Thiết kế giao diện phần mềm.
* Lập trình xử lý các màn hình ứng dụng.
* Viết báo cáo đồ án, in và đóng cuốn theo biểu mẫu quy định.

**Nội dung và yêu cầu đã được thông qua Bộ môn.**

*Tp.HCM, ngày … tháng … năm 2021*

|  |  |
| --- | --- |
| **TRƯỞNG BỘ MÔN**  *(Ký và ghi rõ họ tên)*  ***ThS. Phạm Văn Đăng*** | **GIẢNG VIÊN GIẢNG DẠY**  *(Ký và ghi rõ họ tên)*  ***ThS. Giang Hào Côn*** |

LỜI CẢM ƠN

Dòng viết đầu tiên chính là lời cảm ơn gửi đến thầy Giang Hào Côn, người mà đã tận tình quan tâm, hướng dẫn, cũng như sự giúp đỡ chân thành, những lời góp ý đầy chi tiết để em có thể hoàn thành một sản phẩm một cách tốt nhất.

Và cũng chính vì sự chân thành, tận tụy của thầy giúp đỡ không chỉ em, mà các bạn cùng lớp cũng được thầy chỉ dẫn, đưa ra những ưu điểm và khuyết điểm để có thể cải thiện sản phẩm cũng như kỹ năng cá nhân của mỗi người.

Cuối cùng là lời nói cảm ơn sâu sắc nhất một lần nữa gửi đến thầy Côn đã cùng em và các bạn đi hết chặng đường tuy không dài nhưng nói ngắn thì cũng không chính xác, nó đủ để cảm nhận được sự yêu nghề, tính chuyên nghiệp trong ngành giáo dục của thầy, đặc biệt nhất đó chính là lòng yêu thương sự tận tình giúp đỡ mỗi khi sinh viên cần.

Em cũng chân thành gửi lời cảm ơn đến các Thầy Cô khoa Công nghệ thông tin trường Đại học Nguyễn Tất Thành đã giảng dạy, hướng dẫn, trang bị các kiến thức cho em trong thời gian học vừa qua, từ các kiến thức cơ bản đến các vấn đề chuyên sâu.

Em xin gửi lời cảm ơn đến các anh, chị, bạn bè, đồng nghiệp bằng nhiều hình thức khác nhau đã giúp đỡ em trong quá trình học tập tại trường cũng như trong thời gian hoàn thành đồ án lần này.

Đặc biệt là xin gửi lời cảm ơn đến cha mẹ và anh chị em trong gia đình đã động viên tinh thần cũng như chia sẻ những khó khăn trong những năm qua

Sinh viên thực hiện

**PHẠM TUẤN KIỆT**

LỜI MỞ ĐẦU

Ngày nay, ngành công nghệ thông tin đang phát triển vô cùng mạnh mẽ. Những thành tựu của ngành CNTT được ứng dụng ở mọi lĩnh vực và dần trở thành một phần quan trọng trong cuộc sống hiện đại. Trong số những thành tựu mà CNTT mang lại, việc quản lý thông tin là một ứng dụng cần thiết và mang lại nhiều lợi ích. Nó giúp cho công việc quản lý của các công ty, xí nghiệp, trường học trở nên nhẹ nhàng hơn, tiết kiệm được thời gian, tiền bạc, nhân công và có độ chính xác rất cao.

Tại các trường đại học vấn đề quản lý chỗ ở của sinh viên là một trong những vấn đề thiết thực và cấp bách. Một sinh viên muốn đạt được thành quả cao trong học tập thì cần có một môi trường sống và học tập tốt. Thực tế ở các trường đại học, việc quản lý chỗ ở hầu hết cũng chỉ quản lý trên giấy tờ, sổ sách. Vì vậy việc áp dụng CNTT, đưa máy tính vào trợ giúp quản lý ký túc xá sinh viên là một nhu cầu thiết yếu, nó giúp cho người quản lý thu hẹp không gian lưu trữ, tránh được thất lạc dữ liệu, xử lý nhanh một khối lượng lớn về thông tin, tra cứu tìm kiếm một cách nhanh chóng với độ chính xác cao, tiết kiệm được nhiều thời gian và công sức không chỉ đối với người quản lý mà còn đối với cả gia đình và bản thân sinh viên.

Trước thực trạng đó, cùng với yêu cầu môn học, em quyết định chọn đề tài “TÌM HIỂU VỀ ANGULAR ĐỂ XÂY DỰNG HỆ THỐNG WEBSITE QUẢN LÝ KÝ TÚC XÁ SINH VIÊN CHO TRƯỜNG ĐẠI HỌC NTT”. Với đề tài và môn học này, em xin chân thành cảm ơn sự giúp đỡ tận tình của thầy Giang Hào Côn. Song, do còn nhiều  hạn chế đề tài trong quá trình xây dựng không tránh khỏi những thiếu sót. Rất mong được thầy cô đóng góp ý kiến để website ngày càng hoàn thiện và được đưa vào sử dụng.

NHẬN XÉT CỦA GIẢNG VIÊN

|  |  |
| --- | --- |
|  | TPHCM, Ngày …… tháng …… năm **Giảng viên hướng dẫn** |

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc92229209)

[LỜI MỞ ĐẦU ii](#_Toc92229210)

[NHẬN XÉT CỦA GIẢNG VIÊN iii](#_Toc92229211)

[MỤC LỤC iv](#_Toc92229212)

[MỤC LỤC HÌNH ẢNH vi](#_Toc92229213)

[MỤC LỤC BẢNG BIỂU vii](#_Toc92229214)

[DANH MỤC TỪ VIẾT TẮT viii](#_Toc92229215)

[CHƯƠNG 1: GIỚI THIỆU 1](#_Toc92229216)

[1.1. Lý do chọn đề tài: 1](#_Toc92229217)

[1.2. Mục tiêu nghiên cứu: 1](#_Toc92229218)

[1.3. Đối tượng nghiên cứu: 1](#_Toc92229219)

[1.4. Phạm vi nghiên cứu: 2](#_Toc92229220)

[1.5. Bố cục đề tài: 2](#_Toc92229221)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 3](#_Toc92229222)

[2.1. Giới thiệu Angular: 3](#_Toc92229223)

[2.2. Components – Modules – Binding – Templates: 4](#_Toc92229224)

[2.2.1. Components: 4](#_Toc92229225)

[2.2.2. Modules: 9](#_Toc92229226)

[2.2.3. Binding: 10](#_Toc92229227)

[2.2.4. Templates: 13](#_Toc92229228)

[2.3. Directives – Pipes – Routing : 14](#_Toc92229229)

[2.3.1. Directives 14](#_Toc92229230)

[2.3.2. Pipes: 15](#_Toc92229231)

[2.3.3. Routing: 17](#_Toc92229232)

[2.4. Services – Http Client: 17](#_Toc92229233)

[2.4.1. Services: 17](#_Toc92229234)

[2.4.2. Http Client: 18](#_Toc92229235)

[2.5. Forms: 18](#_Toc92229236)

[2.5.1. Template-Driven Form: 19](#_Toc92229237)

[2.5.2. Model Driven Form: 20](#_Toc92229238)

[CHƯƠNG 3: PHÂN TÍCH, THIẾT KẾ VÀ CÀI ĐẶT 23](#_Toc92229239)

[3.1. Phân tích và đặc tả yêu cầu: 23](#_Toc92229240)

[3.1.1. Phân tích hiện trạng 23](#_Toc92229241)

[3.1.2. Xác định yêu cầu 23](#_Toc92229242)

[3.1.2.1. Yêu cầu chức năng: 23](#_Toc92229243)

[3.1.2.2. Yêu cầu tin học: 23](#_Toc92229244)

[3.1.3. Phân tích 24](#_Toc92229245)

[3.1.3.1. Sơ đồ các chức năng của hệ thống: 24](#_Toc92229246)

[3.1.3.2. Sơ đồ dữ liệu: 24](#_Toc92229247)

[3.2. Thiết kế và cài đặt: 25](#_Toc92229248)

[3.2.1 Thiết kế dữ liệu 25](#_Toc92229249)

[3.2.1.1 Danh sách các quan hệ: 25](#_Toc92229250)

[3.2.1.2 Mô hình dữ liệu quan hệ: 30](#_Toc92229251)

[3.2.1.3 Các ràng buộc toàn vẹn 30](#_Toc92229252)

[3.2.2. Thiết kế xử lý 31](#_Toc92229253)

[3.2.2.1 Sơ đồ liên kết các giao diện: 31](#_Toc92229254)

[3.2.2.2 Hệ thống các giao diện. 31](#_Toc92229255)

[KẾT LUẬN 36](#_Toc92229256)

[TÀI LIỆU THAM KHẢO 37](#_Toc92229257)

MỤC LỤC HÌNH ẢNH

[Hình 2.1: Hình AppComponent 5](#_Toc92228938)

[Hình 2.2: Hình ví dụ về @Input 7](#_Toc92228939)

[Hình 2.3: Luồng hoạt động của @Input 7](#_Toc92228940)

[Hình 2.4: Hình ví dụ về @Output 8](#_Toc92228941)

[Hình 2.5: Luồng hoạt động của @Output 9](#_Toc92228942)

[Hình 2.6: File app.module.ts 10](#_Toc92228943)

[Hình 2.7: Hình minh họa cách sử dụng của EventBinding 11](#_Toc92228944)

[Hình 2.8: Hình minh họa cách sử dụng của Two-way data binding 12](#_Toc92228945)

[Hình 2.9: Hình minh họa cách sử dụng của Interpolation 13](#_Toc92228946)

[Hình 2.10: Hình minh họa cách sử dụng của Property binding 13](#_Toc92228947)

[Hình 2.11: Cách lấy dữ liệu bằng ngModel 14](#_Toc92228948)

[Hình 2.12: Hình ví dụ về UpperCase và LowerCase 16](#_Toc92228949)

[Hình 2.13: Hình ví dụ về Slice 16](#_Toc92228950)

[Hình 2.14: Routing 17](#_Toc92228951)

[Hình 2.15: Cách hoạt động của Template-Driven Form 19](#_Toc92228952)

[Hình 2.16: Ví dụ về Template-Driven Form 20](#_Toc92228953)

[Hình 2.17: Cách hoạt động của Model Driven Form 21](#_Toc92228954)

[Hình 3.1: Sơ đồ use case tổng quát 24](#_Toc92228955)

[Hình 3.2: Sơ đồ class diagram 25](#_Toc92228956)

[Hình 3.3: Sơ đồ quan hệ dữ liệu 30](#_Toc92228957)

[Hình 3.4: Sơ đồ liên kết hệ thống 31](#_Toc92228958)

[Hình 3.5: Giao diện trang đăng ký thuê phòng 31](#_Toc92228959)

[Hình 3.6: Giao diện trang quản lý khoa 32](#_Toc92228960)

[Hình 3.7: Giao diện trang quản lý lớp 32](#_Toc92228961)

[Hình 3.8: Giao diện trang quản lý sinh viên 33](#_Toc92228962)

[Hình 3.9: Giao diện trang quản lý phòng 33](#_Toc92228963)

[Hình 3.10: Giao diện trang quản lý nhân viên 34](#_Toc92228964)

[Hình 3.11: Giao diện trang danh sách đăng ký thuê 34](#_Toc92228965)

[Hình 3.12: Giao diện trang quản lý hợp đồng 35](#_Toc92228966)

[Hình 3.13: Giao diện trang quản lý tiền điện nước 35](#_Toc92228967)

MỤC LỤC BẢNG BIỂU

[Bảng 3.1: Bảng khoa 25](#_Toc92198610)

[Bảng 3.2: Bảng lop 26](#_Toc92198611)

[Bảng 3.3: Bảng sinhvien 26](#_Toc92198612)

[Bảng 3.4: Bảng khunha 27](#_Toc92198613)

[Bảng 3.5: Bảng phòng 27](#_Toc92198614)

[Bảng 3.6: Bảng hopdong 28](#_Toc92198615)

[Bảng 3.7: Bảng hddiennuoc 29](#_Toc92198616)

[Bảng 3.8: Bảng nhanvien 29](#_Toc92198617)

[Bảng 3.9: Bảng ràng buộc toàn vẹn 30](#_Toc92198618)

DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Giải thích |
| CSDL | Cơ sở dữ liệu |  |
| HTML | Hyper Text Markup Language |  |
| MVC | **Model – View – Controller** |  |
| CSS | Cascading Style Sheets |  |
| SQL | Structured Query Language |  |
| PDO | PHP Data Objects |  |
| ASP | Active Server Pages |  |
| sp | Sản phẩm |  |

# GIỚI THIỆU

## 1.1. Lý do chọn đề tài:

Tại các trường đại học vấn đề quản lý chỗ ở của sinh viên là một trong những vấn đề thiết thực và cấp bách. Một sinh viên muốn đạt được thành quả cao trong học tập thì cần có một môi trường sống và học tập tốt. Thực tế ở các trường đại học, việc quản lý chỗ ở hầu hết cũng chỉ quản lý trên giấy tờ, sổ sách. Vì vậy việc áp dụng CNTT, đưa máy tính vào trợ giúp quản lý ký túc xá sinh viên là một nhu cầu thiết yếu, nó giúp cho người quản lý thu hẹp không gian lưu trữ, tránh được thất lạc dữ liệu, xử lý nhanh một khối lượng lớn về thông tin, tra cứu tìm kiếm một cách nhanh chóng với độ chính xác cao, tiết kiệm được nhiều thời gian và công sức không chỉ đối với người quản lý mà còn đối với cả gia đình và bản thân sinh viên.

Trước thực trạng đó, cùng với yêu cầu môn học, em quyết định chọn đề tài “TÌM HIỂU VỀ ANGULAR ĐỂ XÂY DỰNG HỆ THỐNG WEBSITE QUẢN LÝ KÝ TÚC XÁ SINH VIÊN CHO TRƯỜNG ĐẠI HỌC NTT”. Nhằm tạo ra website giúp việc quản lý dễ dàng và tránh thất lạc giấy tờ.

## 1.2. Mục tiêu nghiên cứu:

* Tập trung vào việc thu thập các thông tin, tài liệu liên quan tới cấu trúc của hệ thống và các hoạt động của hệ thống nhằm xác định một số vấn đề trước khi bắt đầu xây dựng, phát triển một dự án.
* Tiếp cận với nghiệp vụ chuyên môn, môi trường hoạt động.
* Giúp nhân viên dễ dàng quản lý các phòng, các khu nhà, duyệt các đơn xin thuê phòng. Giúp sinh viên đăng ký xin thuê phòng dễ dàng.
* Giúp cuộc sống trở nên dễ dàng dàng, tiết kiệm thời gian hơn.
* Cung cấp giao diện trực quan cho người cho người sử dụng.
* Mang đến sự tiện lợi cho người dùng.

1.3. Đối tượng nghiên cứu:

* Nhân viên
* Sinh viên
* Hóa đơn điện nước
* Hợp đồng thuê
* Phòng
* Đăng ký thuê phòng

1.4. Phạm vi nghiên cứu:

Đề tài tập trung vào việc xây dựng các trang quản lý các đối tượng nghiên cứu ở trên. Ngoài ra còn xây dựng chức đăng ký thuê phòng giúp sinh viên có thể đăng ký thuê dễ dàng.

1.5. Bố cục đề tài:

Bố cục đề tài gồm 3 chương, kết luận, tài liệu tham khảo:

* Chương 1: Giới thiệu.
* Chương 2: Cơ sở lý thuyết.
* Chương 3: Phân tích, thiết kế và cài đặt.
* Kết luận.
* Tài liệu tham khảo.

# CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu Angular:

Angular là một nền tảng hay JavaScript Framework được phát triển để xây dựng các Single Page Application (SPA) sử dụng HTML, JavaScript và TypeScript … Nó triển khai chức năng cốt lõi và các chức năng tùy chọn như một tập hợp các thư viện TypeScript mà các lập trình viên import vào các ứng dụng của mình.

Đây là một sản phẩm được viết bởi Misko Hevery và một người bạn của anh là Adam Abrons. Sau đó, chính thức “ra mắt” vào 20/10/2010. Hiện tại, sản phẩm này vẫn đang được Google duy trì và phát triển.

Phiên bản Angular đầu tiên được phát triển vào năm 2009 được gọi là Angular 1 hay còn được biết đến với cái tên AngularJS. Sau đó đến các phiên bản Angular 2, 3, 4, 5,…, 11 cho đến cuối cùng, phiên bản hiện tại, Angular 12, được phát hành vào ngày 12/05/2021.

Mỗi phiên bản Angular tiếp theo được cải thiện trên phiên bản tiền nhiệm của nó, sửa lỗi, giải quyết các vấn đề và đáp ứng sự phức tạp ngày càng tăng của các nền tảng hiện tại. Để bắt đầu sử dụng Angular, người dùng cần nắm các kiến thức cơ bản về: HTML, CSS, JavaScript, TypeScript, Document Object Model (DOM).

* **Ưu điểm của Angular:**
* AngularJS được đánh giá là giải pháp dành cho các Single Page Application làm việc dễ dàng.
* Code Front end thường rất thân thiện nhờ khả năng Binding data lên trên các nền tảng HTML để có thể tạo ra mọi thao tác rất tuyệt.
* Bạn có thể dễ dàng Unit test
* Có thể tái sử dụng component dễ dàng hơn
* Hỗ trợ cho các lập trình viên có thể viết code được ít hơn với nhiều chức năng hơn.
* Bạn có thể chạy AngularJS trên nhiều loại trình duyệt khác nhau lẫn PC hoặc mobile.
* **Nhược điểm của Angular:**
* Không an toàn: Thông thường, bản chất của Angular là một trong những Framework Front End, mà Front end này thường vốn không thể bảo mật bằng Back-end. Chính vì vậy, khi sử dụng API thì bạn cần xây dựng cho một hệ thống kiểm tra dữ liệu sao cho việc trả về được tốt nhất.
* Bị hạn chế về trình duyệt: Vài trình duyệt được tích hợp tính năng Disable Javascript, mà Angular được lập trình bằng ngôn ngữ này nên không thể sử dụng trên trình duyệt đó.

2.2. Components – Modules – Binding – Templates:

2.2.1. Components:

* Một Angular Component bao gồm:
* Một mẫu (template) cái mà sẽ tạo ra giao diện (layout view) cho người dùng, template này được tạo ra từ HTML. Trên trang HTML này, chúng ta sẽ sử dụng Angular Binding và Directives để tăng sức mạnh cho View.
* Component sẽ có một Class để chúng ta liên kết với View. Các Class này được tạo ra bằng TypeScript. Class có chứa các thuộc tính và phương thức, cũng tương tự như định nghĩa Class của lập trình hướng đối tượng.
* Một phần không thể thiếu khác của Component là Metadata, nó cung cấp thông tin bổ xung cho Angular Component và giúp định nghĩa Class này như một Angular Component. Metadata được định nghĩa bởi một Decorator. Một decorator có thể hiểu đơn giản là phương thức thêm Metadata vào Class.
* Tạo một Component Class:
* Một class là một cấu trúc cho phép tạo những thuộc tính(properties) được xác định bởi những kiểu dữ liệu (data type) và những phương thức (method) cung cấp những chức năng.
* Định nghĩa một class sử dụng một từ khoá “class” và theo đó là tên của class. Thông thường Angular quy ước đặt tên mỗi component với một tên đặc trưng. Sau đó nối các thành phần như hậu tố “Component”.
* Khi khởi tạo 1 ứng dụng Angular thì **Angular CLI** sẽ giúp chúng ta khởi tạo ra 1 Component có tên là **App Component** trong thư mục src/app:



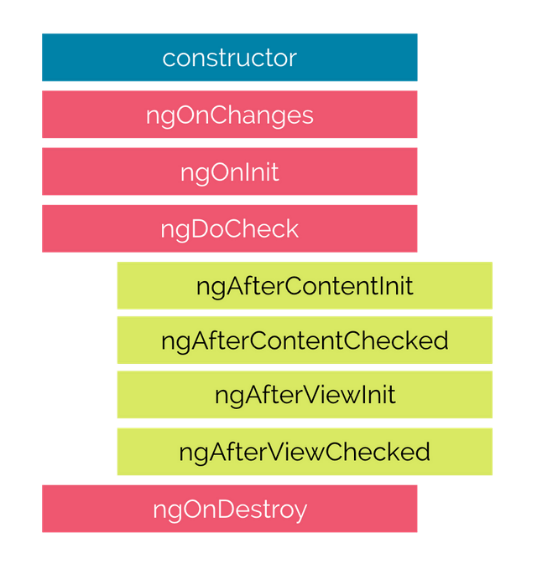
Hình 2.1: Hình AppComponent

* **AppComponent** là thành phần cha của ứng dụng. Tất cả các thành phần mới tạo ra sau này đều là thành phần con của **AppComponent**.
* Để tạo component mới, ta sử dụng lệnh **Angular CLI**:

ng generate component <component-name>

* Hoặc bằng cách ngắn hơn

ng g c <component-name>

* Vòng đời của Component:

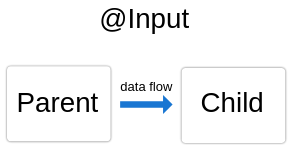
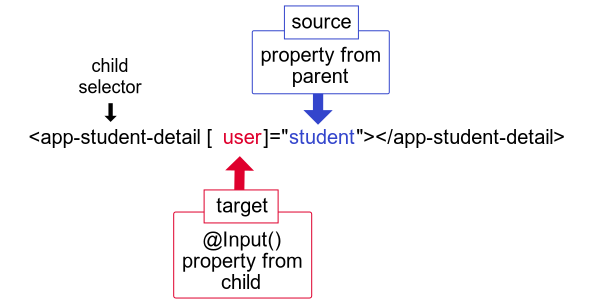
Vòng đời của một component Angular hỗ trợ các Interface để tạo nên sự tuần tự trong quá trình hoạt động. Một vòng đời của Component sẽ có thứ tự như sau: ngOnChanges, ngOnInit, ngDoCheck ngAfterContentInit, ngAfterContentChecked, ngAfterViewInit, ngAfterViewChecked, OnDestroy.

* Truyền dữ liệu giữa các Component:
* Angular cung cấp cho chúng ta 2 decorator để truyền dữ liệu giữa các Component là @**Input** và @**Output**.
* Decorator là một cú pháp khai báo đặc biệt, không bao giờ đứng độc lập mà luôn được gắn kèm với một khai báo class, method, property hoặc accessor. Mọi người có thể thấy trong mỗi Component thì @**Component** cũng là 1 decorator.
* @**Input**: dùng để truyền dữ liệu từ component cha sang component con. Vd:





Hình 2.2: Hình ví dụ về @Input

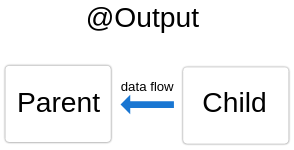
Hình 2.3: Luồng hoạt động của @Input

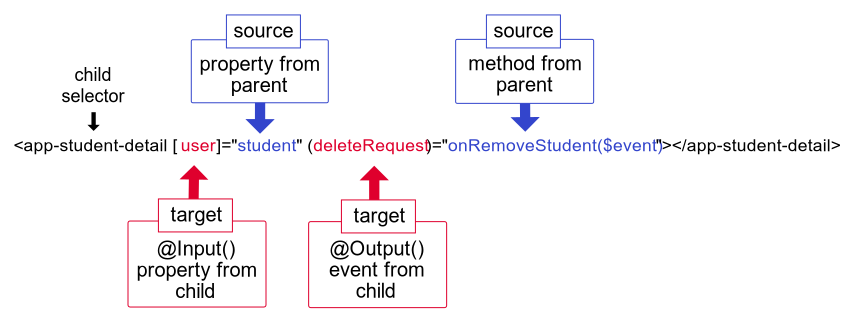
* @**Output**: Bắt sự kiện cho eventEmitter được truyền vào component. Tức là mỗi khi có sự kiện xảy ra ở trên View như là onClick, onChange, onMouseMove...thì @Output sẽ trả ngược lại event cho component cha





Hình 2.4: Hình ví dụ về @Output





Hình 2.5: Luồng hoạt động của @Output

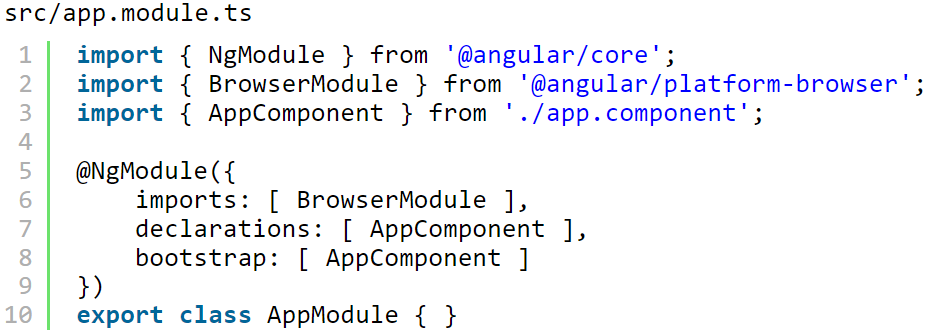
2.2.2. Modules:

Các ứng dụng Angular có dạng module và Angular có hệ thống module riêng được gọi là NgModule. NgModule là vùng chứa cho một khối mã liên kết dành riêng cho miền ứng dụng, quy trình làm việc hoặc tập hợp các chức năng có liên quan chặt chẽ.

Chúng có thể chứa các component, nhà cung cấp dịch vụ và các tệp mã khác có phạm vi được xác định bởi NgModule chứa. Chúng có thể import chức năng được export từ ​​các NgModule khác và export cho các NgModule khác sử dụng.

Một NgModule được định nghĩa bởi một lớp với decorator @NgModule() . Decorator @NgModule() là một function nhận một đối tượng siêu dữ liệu để mô tả cho module. Các thuộc tính quan trọng nhất của @NgModule() là:

* + declarations: Các component, directive và pipe thuộc về NgModule này.
  + exports: Tập hợp con các khai báo sẽ hiển thị và có thể sử dụng được trong các component template của các NgModule khác.
  + imports: Các module khác có các lớp đã export là cần thiết cho component template được khai báo trong NgModule này.
  + providers: Trình cung cấp service mà NgModule này đóng góp vào bộ sưu tập dịch vụ toàn cục; chúng có thể truy cập được trong tất cả các phần của ứng dụng. (Bạn cũng có thể chỉ định các nhà cung cấp ở cấp component.)
  + bootstrap: View chính của ứng dụng, được gọi là component gốc, nó chứa tất cả các view khác của ứng dụng. Chỉ có NgModule gốc mới nên thiết lập giá trị thuộc tính bootstrap.



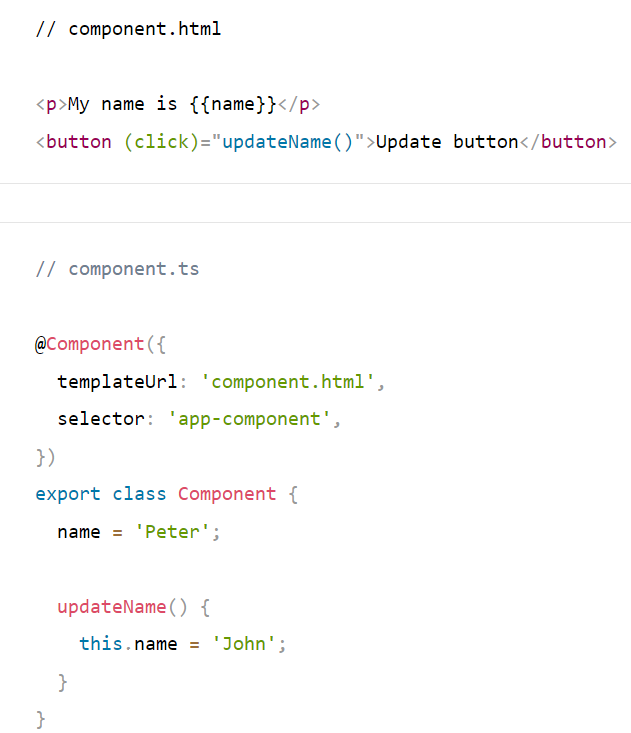
Hình 2.6: File app.module.ts

2.2.3. Binding:

**Data binding** là 1 kĩ thuật quan trọng và mạnh mẽ trong khía cạnh phát triển ứng dụng. Nó hỗ trợ tương tác giữa component và view riêng của nó. Giúp ứng dụng của chúng ta trở nên linh hoạt hơn.

Trong Angular có 4 dạng chính data binding:

* **Event binding:**  thực hiện gửi giá trị từ Component cho Template và gửi những giá trị thay đổi của Tempate lại cho Component. Điều đó có nghĩa là những giá trị thay đổi trên Template sẽ tự động được thay đổi trên Component.



Hình 2.7: Hình minh họa cách sử dụng của EventBinding

* **Two-way data binding:** Sử dụng [()] thực hiện gửi giá trị từ Component cho Template và gửi những giá trị thay đổi của Tempate lại cho Component. Điều đó có nghĩa là những giá trị thay đổi trên Template sẽ tự động được thay đổi trên Component.





Hình 2.8: Hình minh họa cách sử dụng của Two-way data binding

* **Interpolation:** Sử dụng cặp {{}} để thực hiện render ra giá trị khi thực hiện gắn kết giữa Data và template của component. Cách gắn kết này được thực hiện theo cơ chế một chiều (Chỉ đọc thông tin), nó tiện lợi khi lập trình viên muốn hiển thị dữ liệu đơn giản.



Hình 2.9: Hình minh họa cách sử dụng của Interpolation

* **Property binding:** Cơ chế gắn kết thông qua thuộc tính. Cách thể hiện của đặc tính này là thông qua []. Đây cũng là cơ chế gắn kết 1 chiều.

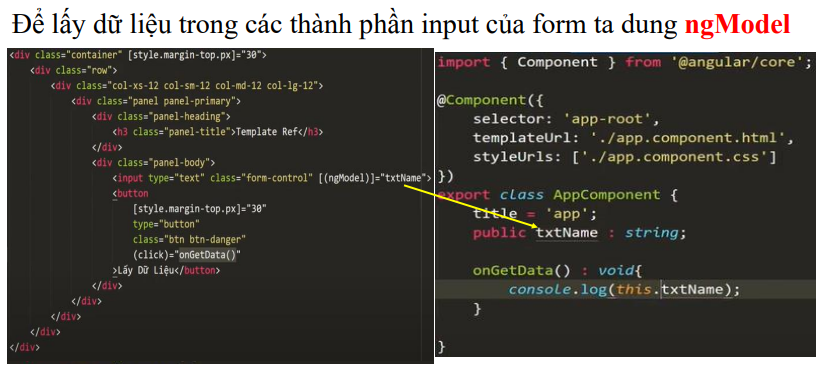


Hình 2.10: Hình minh họa cách sử dụng của Property binding

2.2.4. Templates:

Một Template Reference Variables là một tham chiếu đến một phần tử DOM hoặc Directive trong template. Sử dụng biến tham chiếu, ta có thể truy cập vào các giá trị thuộc tính phần tử DOM. Để thao tác với template Variable có 2 cách:

* + Tại Template: dùng #name hoặc ref-name
  + Tại Component: dùng @ViewChild(“name1”) name2: ElementRef



Hình 2.11: Cách lấy dữ liệu bằng ngModel

2.3. Directives – Pipes – Routing :

2.3.1. Directives

Directives là một đối tượng giúp chúng ta dễ dàng thay đổi một đối tượng khác và cách áp dụng rất đơn giản và linh hoạt.

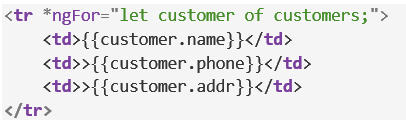
Chúng ta sử dụng Directive để thao tác, cập nhật, thêm, xóa các thành phần giao diện trên web. (Hay nói cách khác chúng ta thao tác với các element của DOM. Các nút, ảnh, text box, table mà ta thấy trên website được một đối tượng DOM quản lý. DOM có nhiệm vụ vẽ ra các đối tượng này trên website để ta có thể thấy được).

Angular Directive được chia là 3 loại gồm:

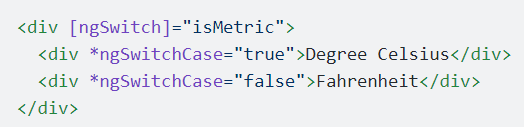
* Components **directives**: Không có nghi ngờ gì khi gọi component là directive cũng được, vì rõ ràng là component cho phép định nghĩa selector và gọi ra như một thẻ html tag (<component-name></component-name>)
* Structural directives: Là directive cấu trúc, dùng để vẽ html, hiển thị data lên giao diện html.
* Attribute directives: Thêm các thuộc tính động cho element html, ví dụ ngStyle
  + Structural directives:
* NgIf…else: có tác dụng kiểm tra điều kiện ngay ở html, cũng hỗ trợ các toán tử logic, giống như một câu lệnh if thông thường trong JavaScript.



* NgFor: để duyệt qua các tập hợp dữ liệu như array, list. Sau đó tạo các thành phần web tương ứng với mỗi vòng lặp.



* NgSwitch: dùng để quyết định thêm hoặc xóa các thành phần của web phụ thuộc vào điều kiện có thoả mãn hay không. Chúng ta thường dùng chung với ngSwitchCase và ngSwitchDefault



* + Attribute directives:
* ngModel : được sử dụng cho việc binding 2 chiều.
* ngClass : được sử dụng để thêm hoặc xoá một class của một thành phần web.
* NgStyle: thêm/bớt các style.

2.3.2. Pipes:

Pipes Là một tính năng được xây dựng sẵn từ Angular với mục tiêu nhằm biến đổi dữ liệu trước khi hiển thị ra cho người dung. Một số pipe dựng sẵn trong Angular.

* + **UpperCase và LowerCase**:

Được sử dụng để viết hoa toàn bộ hoặc viết thường toàn bộ dữ liệu. Ví dụ:



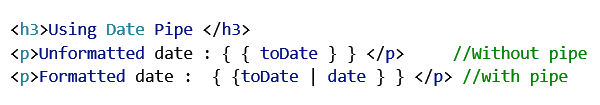
Hình 2.12: Hình ví dụ về UpperCase và LowerCase

* + **Date:**

Chuyển dữ liệu thành định dạng ngày tháng currency: Chuyển dữ liệu thành định dạng tiền tệ.

date\_expression | date[:format]

* date\_expression là đối tượng ngày hoặc là một kiểu số.
* data : tên của pipe sẽ được dùng để định dạng dữ liệu.



* + **Slice:** Cắt dữ liệu của mảng hoặc chuỗi.



Hình 2.13: Hình ví dụ về Slice

* **Decimal và Number:** Dùng để format cho kiểu số và kiểu thập phân
* **Lowercase và uppercase:** Chuyển chuỗi thành kí tự thường và Chuyển chuỗi thành kí tự in hoa
* **Percent:** Chuyển dữ liệu về dạng %
* **Json:** Chuyển dữ liệu về dạng JSON

**Custome Pipe:** Trong quá trình xử lý dữ liệu để hiển thị ra cho người dùng thì các pipe mà Angular hỗ trợ sẵn thì sẽ là chưa đủ so với yêu cầu mà bài toán đưa ra, để có để đáp ứng được yêu cầu đó thì ta có thể tự viết ra các pipe để phù hợp với bài toán.

Cú pháp tạo Pipe:

2.3.3. Routing:

Để tạo được các liên kết trong Angular, ta có thể sử dụng Routing. Routing trong Angular về cơ bản có nghĩa là điều hướng giữa các trang. Với các ứng dụng web thông thường, việc điều hướng theo URL thường sẽ do phía server đảm nhiệm.

Khi tạo project bằng lệnh ng new ten\_project , hệ thống hỏi chúng ta có muốn tạo routing hay không? Nếu nhập yes là có. Có nghĩa là Routing Module được thêm vào file app.module.ts như sau:



Hình 2.14: Routing

2.4. Services – Http Client:

2.4.1. Services:

Angular Service là những đoạn code mà ta có thể sử dụng nhiều lần từ các component khác nhau. Nó có chức năng sử dụng lại. Những đoạn code này sẽ thực hiện một nhiệm vụ cụ thể cho một ý định nào đó.

Chúng ta sử dụng service cho những mục đích:

* + Những nhiệm vụ độc lập của component như ghi log, gọi api từ bên ngoài...
  + Chia sẻ code logic hoặc data để các component có thể dùng chung.

Lợi thế của Service:

* Dễ dàng thực hiện các việc test cho component và service.
* Dễ dàng debugs khi có vấn đề.
* Có thể được sử dụng lại ở nhiều module.

2.4.2. Http Client:

Hầu hết các ứng dụng front-end cần giao tiếp với máy chủ qua giao thức HTTP để tải xuống hoặc tải lên dữ liệu và truy cập các dịch vụ back-end khác. Angular cung cấp một ứng dụng khách HTTP API cho các ứng dụng Angular, lớp dịch vụ HttpClient trong @angular/common/[http](https://angular.io/api/common/http).

Http Client cung cấp các tính năng chính sau đây:

* + Khả năng yêu cầu các đối tượng phản hồi đã nhập.
  + Xử lý lỗi được sắp xếp hợp lý.
  + Tính năng kiểm tra.
  + Yêu cầu và đánh chặn phản hồi.

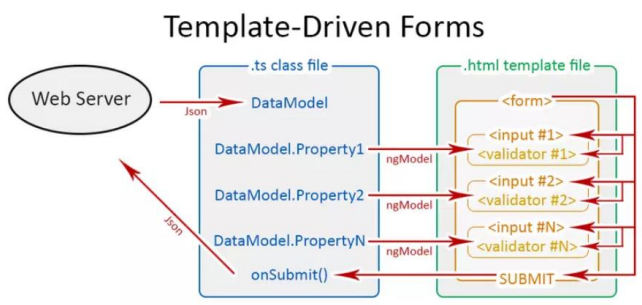
2.5. Forms:

Đối với 1 ứng dụng web thì Form giúp chúng ta có thể thu thập được dữ liệu từ người dùng. Có 2 loại Form trong Angular là Templatedriven form và Model Driven form.

* Template-driven form là phương pháp mà chúng ta sẽ tạo forms dựa vào template. Chúng ta thực hiện việc thêm các directives và hành vi vào template, sau đó Angular sẽ tự động tạo forms để quản lý và sử dụng.
* Model Driven Form hay còn gọi là Reactive Form, phương pháp này tránh việc sử dụng các directive ví dụ như ngModel, required, ..., thay vào đó tạo các Object Model ở trong các Component, rồi tạo ra form từ chúng.

2.5.1. Template-Driven Form:

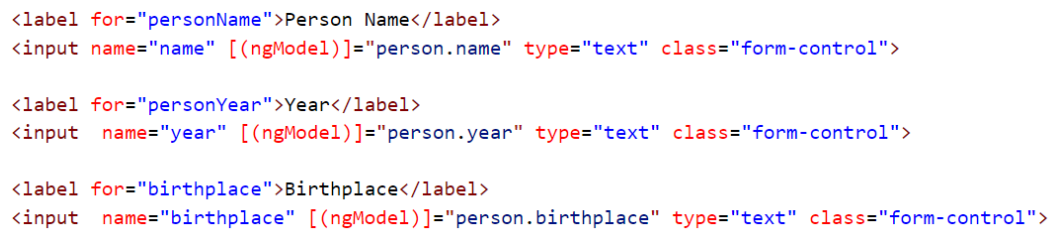
* Luồng hoạt động:
* User thay đổi dữ liệu trong các ô input như là input, checkbox, text area...
* Input Element sẽ gửi đi input event chứa value từ user vừa nhập vào
* Control\_Value\_Accessor sẽ trigger phương thức setValue( ) trên FormControl instance.
* FormControl instance sẽ emit giá trị mới qua valueChanges observable
* Bất cứ subscriber đến valueChanges observable đểu nhận được giá trị mới nhất của control đó.
* Control\_Value\_Accessor cũng gọi phương thức NgModel.viewToModelUpdate( ) để emit ngModelChange event.
* Trong trường hợp chúng ta sử dụng cơ chế two-way binding thì lúc này property trong component sẽ được update value bởi ngModelChange event).
* Khi property trong Component thay đổi thì cơ chế phát hiện thay đổi bắt đầu chạy và control\_Value\_Accessor sẽ update Input Element trong view với giá trị mới nhất của property trong component.



Hình 2.15: Cách hoạt động của Template-Driven Form

Để có thể sử dụng các APIs mà Angular cung cấp cho việc thao tác với Template-driven forms, chúng ta cần import NgModule là FormsModule từ package [*@angular*](https://viblo.asia/u/angular)*/forms* như sau:



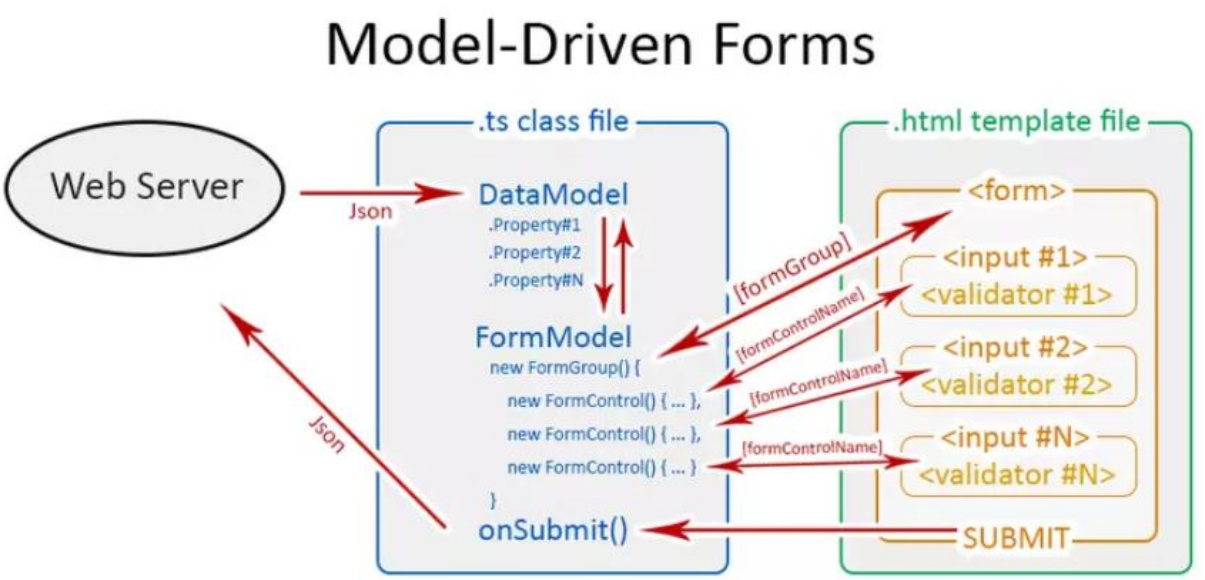


Hình 2.16: Ví dụ về Template-Driven Form

2.5.2. Model Driven Form:

Model Driven Forms hay còn được gọi là Reactive Forms là một phương pháp để tạo form trong Angular, phương pháp này tránh việc sử dụng các directive ví dụ như ngModel, required, etc, thay vào đó tạo các Object Model ở trong các Component, rồi tạo ra form từ chúng. Một điều lưu ý đó là Template-Driven là async còn Reactive là sync.

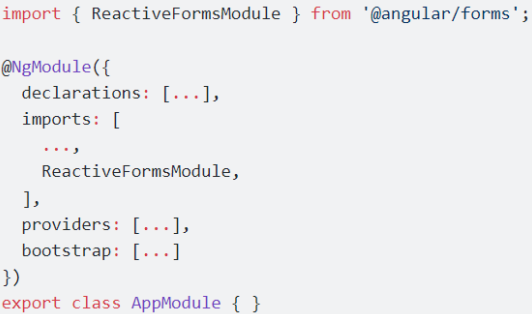
Trong Reactive forms, chúng ta tạo toàn bộ form control tree ở trong code (khởi tạo ngay, khởi tạo trong constructor, hoặc khởi tạo trong ngOnInit), nên có thể dễ dàng truy cập các phần tử của form ngay tức thì.



Hình 2.17: Cách hoạt động của Model Driven Form

* Các thành phần cơ bản:
* AbstractControl là một abstract class cho 3 lớp con form control: FormControl, FormGroup, và FormArray. Nó cung cấp các hành vi, thuộc tính chung cho các lớp con.
* FormControl là đơn vị nhỏ nhất, nó lưu giữ giá trị và trạng thái hợp lệ của một form control. Tương ứng với một HTML form control như input, select.
* FormGroup nó lưu giữ giá trị và trạng thái hợp lệ của một nhóm các đối tượng thuộc AbstractControl – có thể là FormControl, FormGroup, hay FormArray – đây là một dạng composite. Ở level cao nhất của một form trong component của bạn là một FormGroup.
* FormArray nó lưu giữ giá trị và trạng thái hợp lệ của một mảng các đối tượng thuộc AbstractControl giống như FormGroup. Nó cũng là một dạng composite. Nhưng nó không phải là thành phần ở level cao nhất.

Để có thể sử dụng các APIs mà Angular cung cấp cho việc thao tác với Reactive forms, chúng ta cần import ReactiveFormsModule từ package @angular/forms như sau:



# PHÂN TÍCH, THIẾT KẾ VÀ CÀI ĐẶT

3.1. Phân tích và đặc tả yêu cầu:

3.1.1. Phân tích hiện trạng

Ký túc xá của trường có các phòng mỗi phòng có số lượng tối đa mà sinh viên có thể ở, ngoài ra phòng còn phân theo loại (phòng nam, phòng nữ), tình trạng phòng (đầy hay còn chỗ). Phòng còn phân theo khu nhà để quản lý.

Nhân viên sẽ kiểm tra các phòng xem số lượng sinh viên đang ở để cập nhật tình trạng phòng tránh việc đăng ký thuê phòng đã đầy.

Sinh viên khi đăng ký thuê cần cung cấp thông tin như họ và tên, cmnd, số điện thoại, mã sinh viên,….

Sau khi nhận đơn yêu cầu thuê phòng, nhân viên xác nhận thông tin và liên hệ với sinh viên muốn thuê. Khi hai bên thỏa thuận xong, nhân viên sẽ lập hợp đồng thuê phòng bao gồm: mã sinh viên, mã nhân viên lập hợp đồng, ngày bắt đầu ở, ngày kết thúc, ngày lập hợp đồng,…

Mỗi phòng khi có sinh viên thuê sẽ được nhân viên kiểm chỉ số điện và nước đầu khi bắt đầu đền cuối tháng nhân viên phải kiểm tra lại chỉ số diện và nước cuối. Sau khi kiểm nhân viên sẽ lập hóa đơn điện nước tính tổng số tiền điện và nước đã sử dụng.

3.1.2. Xác định yêu cầu

3.1.2.1. Yêu cầu chức năng:

Các chức năng hệ thống cần có:

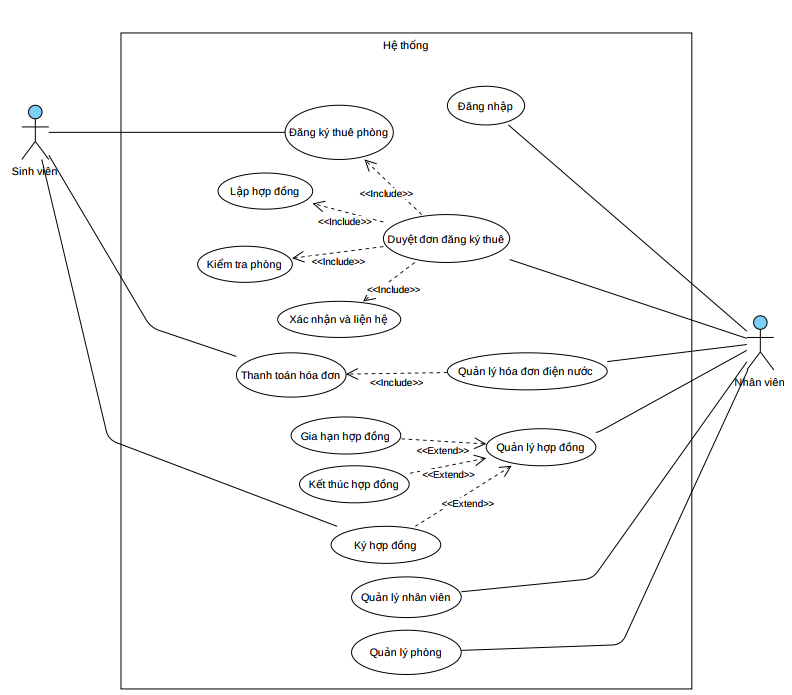
* Quản lý phòng.
* Quản lý nhân viên.
* Quản lý sinh viên.
* Quản lý hóa đơn điện nước.
* Quản lý hợp đồng.
* Duyệt đăng ký thuê phòng.

3.1.2.2. Yêu cầu tin học:

* Cần có Xampp hoặc Wamp để chạy mysql.
* Cần có Nodejs để chạy code backend.
* Visual Studio Code.

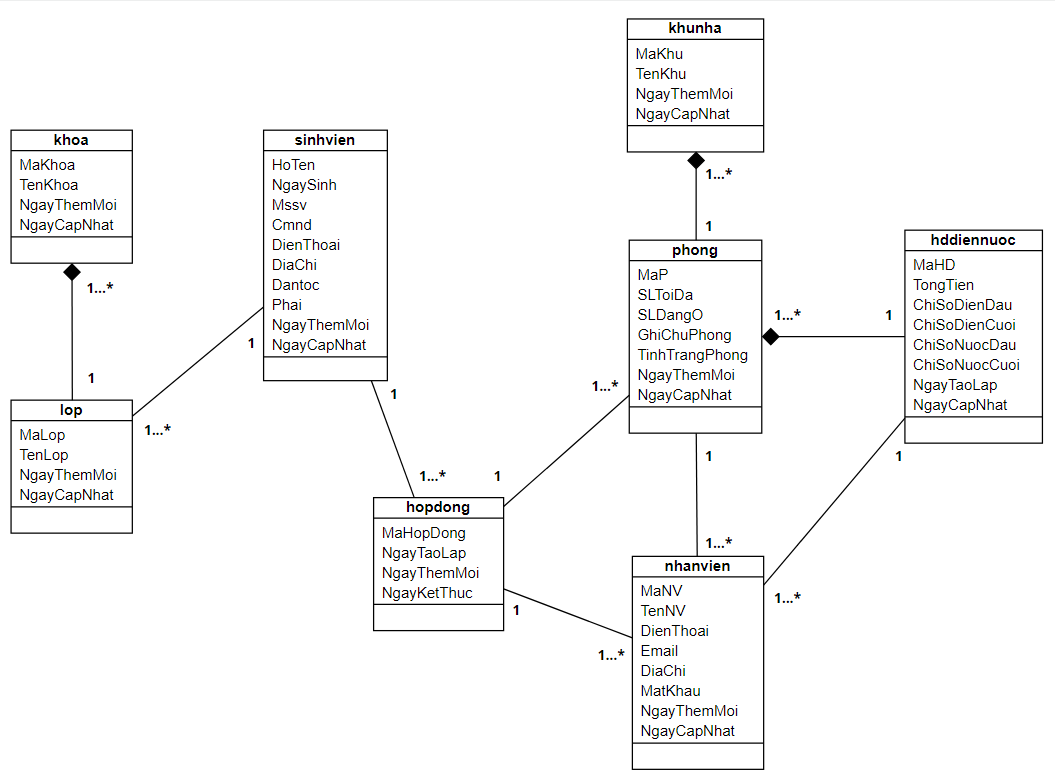
3.1.3. Phân tích

3.1.3.1. Sơ đồ các chức năng của hệ thống:



Hình 3.1: Sơ đồ use case tổng quát

3.1.3.2. Sơ đồ dữ liệu:



Hình 3.2: Sơ đồ class diagram

3.2. Thiết kế và cài đặt:

3.2.1 Thiết kế dữ liệu

3.2.1.1 Danh sách các quan hệ:

|  |  |
| --- | --- |
| **khoa** | |
| **Stt** | **Tên trường** | **Mô tả** | **Kiểu dữ liệu** | **Kích thước** | **Ràng buộc** |
| 1 | MaKhoa | Mã khoa | varchar | 10 | Khóa chính |
| 2 | TenKhoa | Tên khoa | varchar | 100 | Not null |
| 3 | NgayThemMoi | Ngày thêm mới | datetime |  | Not null |
| 4 | NgayCapNhat | Ngày cập nhật | datetime |  | Not null |

Bảng 3.1: Bảng khoa

|  |  |
| --- | --- |
| **lop** | |
| **Stt** | **Tên trường** | **Mô tả** | **Kiểu dữ liệu** | **Kích thước** | **Ràng buộc** |
| 1 | MaLop | Mã lớp | varchar | 10 | Khóa chính |
| 2 | TenLop | Tên lớp | varchar | 100 | Not null |
| 3 | MaKhoa | Mã khoa | varchar | 10 | Khóa ngoại |
| 4 | NgayThemMoi | Ngày thêm mới | datetime |  | Not null |
| 5 | NgayCapNhat | Ngày cập nhật | datetime |  | Not null |

Bảng 3.2: Bảng lop

|  |  |
| --- | --- |
| **sinhvien** | |
| **Stt** | **Tên trường** | **Mô tả** | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | Mssv | Mã khoa | varchar | 10 | | Khóa chính | |
|  | HoTen | Tên khoa | varchar | 100 | | Not null | |
|  | NgaySinh | Ngày sinh | datetime |  | | Not null | |
|  | Cmnd | CMND | varchar | 10 | | Not null | |
|  | DienThoai | Điện thoại | varchar | 10 | | Not null | |
|  | DiaChi | Địa chỉ | varchar | 255 | | Not null | |
|  | DanToc | Dân tộc | varchar | 50 | | Not null | |
|  | Phai | Phái | boolean |  | | Not null | |
|  | MaLop | Mã lớp | varchar | 10 | | Khóa ngoại | |
|  | NgayThemMoi | Ngày thêm mới | datetime |  | | Not null | |
|  | NgayCapNhat | Ngày cập nhật | datetime |  | | Not null | |

Bảng 3.3: Bảng sinhvien

|  |  |  |
| --- | --- | --- |
| **khunha** | | |
| **Stt** | **Tên trường** | **Mô tả** | | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | MaKhu | Mã khu nhà | | varchar | 10 | | Khóa chính | |
|  | TenKhu | Tên khu nhà | | varchar | 10 | | Not null | |
|  | NgayThemMoi | Ngày thêm mới | | datetime |  | | Not null | |
|  | NgayCapNhat | Ngày cập nhật | | datetime |  | | Not null | |

Bảng 3.4: Bảng khunha

|  |  |  |
| --- | --- | --- |
| **phong** | | |
| **Stt** | **Tên trường** | **Mô tả** | | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | MaP | Mã khu nhà | | varchar | 10 | | Khóa chính | |
|  | MaKhu | Tên khu nhà | | varchar | 10 | | Not null | |
|  | MaNV | Mã nhân viên | | varchar |  | | Khóa ngoại | |
|  | SLToiDa | Số lượng tối đa | | int | 6 | | Not null | |
|  | SLDangO | Số lượng đang ở | | int | 6 | | Not null | |
|  | GhiChuPhong | Ghi chú | | varchar |  | | Not null | |
|  | LoaiPhong | Loại phòng | | boolean |  | | Not null | |
|  | TinhTrangPhong | Tình trang | | boolean |  | | Not null | |
|  | NgayThemMoi | Ngày thêm mới | | datetime |  | | Not null | |
|  | NgayCapNhat | Ngày cập nhật | | datetime |  | | Not null | |

Bảng 3.5: Bảng phòng

|  |  |  |
| --- | --- | --- |
| **hopdong** | | |
| **Stt** | **Tên trường** | **Mô tả** | | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | MaHopDong | Mã hợp đồng | | varchar | 10 | | Khóa chính | |
|  | MaNV | Mã nhân viên | | varchar | 10 | | Khóa ngoại | |
|  | Mssv | Mã sinh viên | | varchar | 10 | | Khóa ngoại | |
|  | MaP | Mã phòng | | varchar | 10 | | Khóa ngoại | |
|  | NgayTaoLap | Ngày tạo lập | | datetime |  | | Not null | |
|  | NgayBatDau | Ngày bắt đầu | | datetime |  | | Not null | |
|  | NgayKetThuc | Ngày kết thúc | | datetime |  | | Not null | |

Bảng 3.6: Bảng hopdong

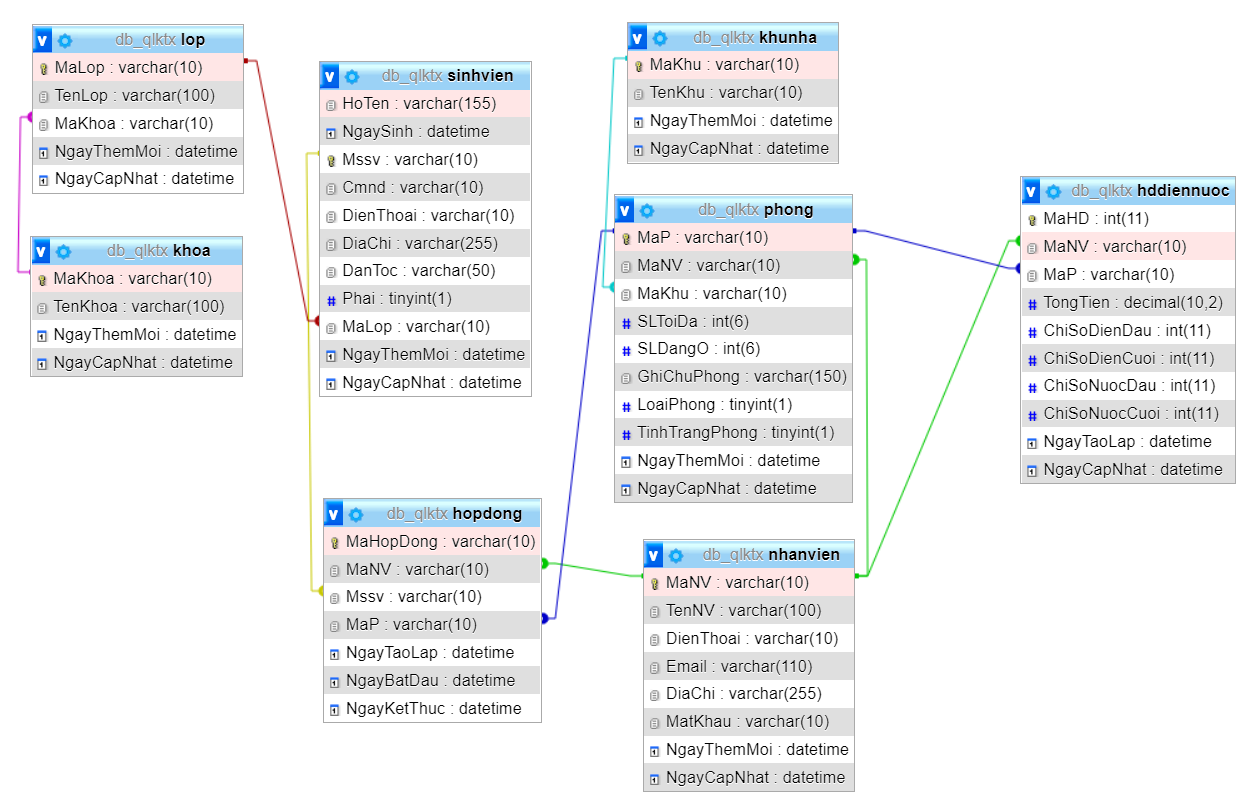
|  |  |  |
| --- | --- | --- |
| **hddiennuoc** | | |
| **Stt** | **Tên trường** | **Mô tả** | | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | MaHD | Mã hợp đồng | | varchar | 10 | | Khóa chính | |
|  | MaNV | Mã nhân viên | | varchar | 10 | | Khóa ngoại | |
|  | MaP | Mã phòng | | varchar | 10 | | Khóa ngoại | |
|  | TongTien | Tổng tiền | | decimal | 10,2 | | Not null | |
|  | ChiSoDienDau | Chỉ số điện đầu | | int | 11 | | Not null | |
|  | ChiSoDienCuoi | Chỉ số điện cuối | | int | 11 | | Not null | |
|  | ChiSoNuocDau | Chỉ số nước đầu | | int | 11 | | Not null | |
|  | ChiSoNuocCuoi | Chỉ số nước cuối | | int | 11 | | Not null | |
|  | NgayTaoLap | Ngày tạo lập | | datetime |  | | Not null | |
|  | NgayCapNhat | Ngày cập nhật | | datetime |  | | Not null | |

Bảng 3.7: Bảng hddiennuoc

|  |  |  |
| --- | --- | --- |
| **nhanvien** | | |
| **Stt** | **Tên trường** | **Mô tả** | | **Kiểu dữ liệu** | | **Kích thước** | | **Ràng buộc** | |
|  | MaNV | Mã nhân viên | | varchar | 10 | | Khóa chính | |
|  | TenNV | Mã phòng | | varchar | 10 | | Not null | |
|  | DienThoai | Điện thoại | | varchar | 10 | | Not null | |
|  | Email | Email | | int | 11 | | Not null | |
|  | DiaChi | Địa chỉ | | int | 11 | | Not null | |
|  | MatKhau | Mật khẩu | | int | 11 | | Not null | |
|  | NgayThemMoi | Ngày tạo lập | | datetime |  | | Not null | |
|  | NgayCapNhat | Ngày kết thúc | | datetime |  | | Not null | |

Bảng 3.8: Bảng nhanvien

3.2.1.2 Mô hình dữ liệu quan hệ:



Hình 3.3: Sơ đồ quan hệ dữ liệu

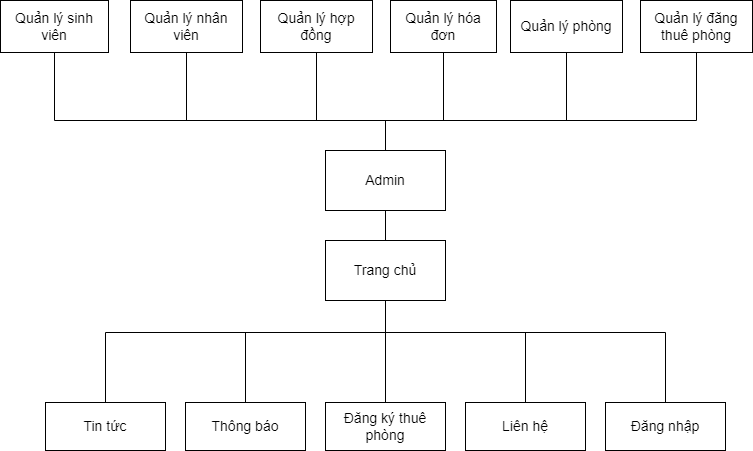
3.2.1.3 Các ràng buộc toàn vẹn

|  |  |
| --- | --- |
| **Tên bảng** | **Ràng buộc toàn vẹn** |
| hopdong | ALTER TABLE hopdong  ADD CONSTRAINT check\_ngayketthuc CHECK (**NgayKetThuc** > **NgayBatDau**) |
| hddiennuoc | ALTER TABLE hddiennuoc  ADD CONSTRAINT check\_sodiencuoi CHECK (**ChiSoDienCuoi** > **ChiSoDienDau**),  ADD CONSTRAINT check\_sonuoccuoi CHECK (**ChiSoNuocCuoi** > **ChiSoNuocDau**) |
|  |  |

Bảng 3.9: Bảng ràng buộc toàn vẹn

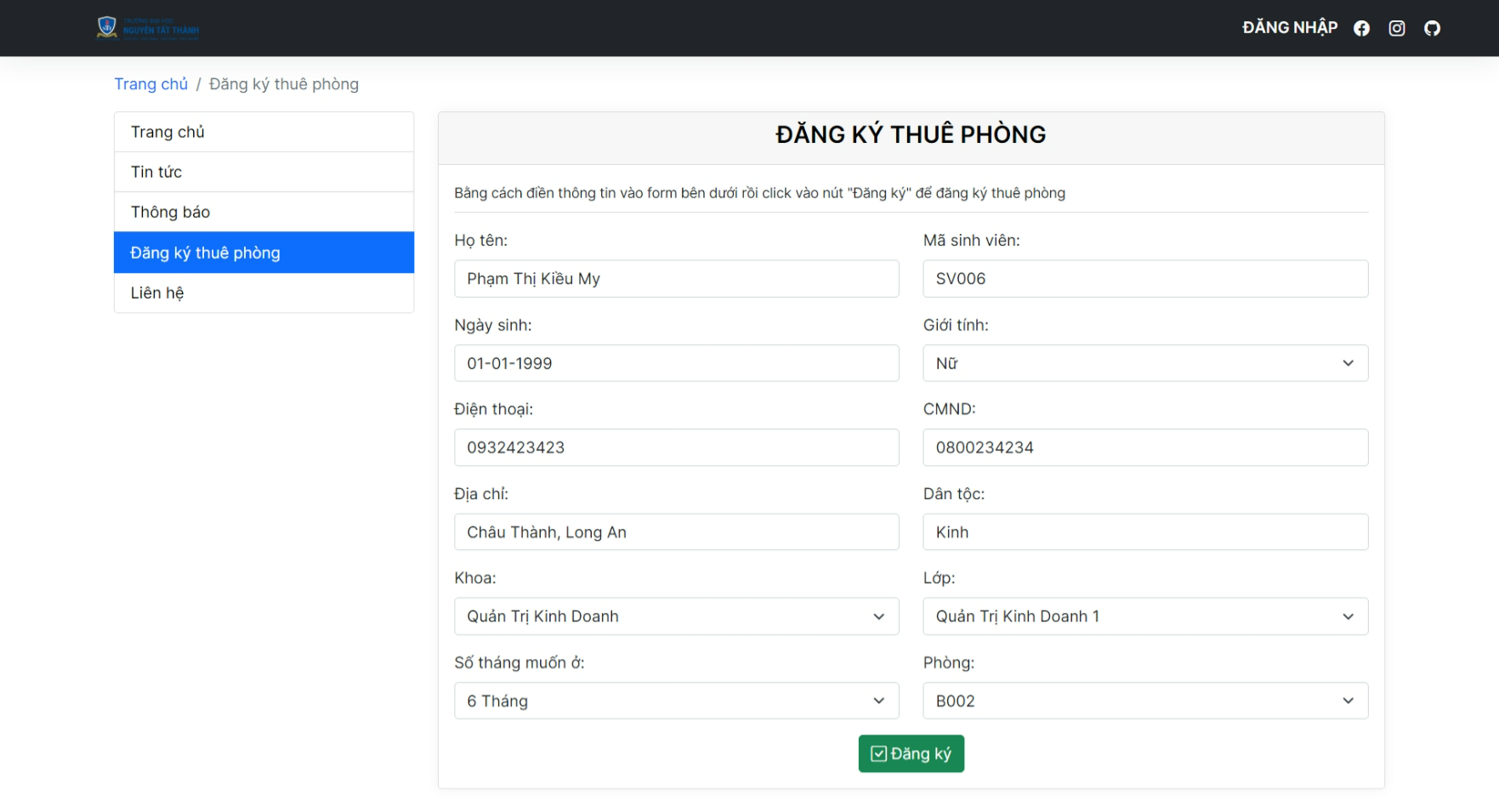
3.2.2. Thiết kế xử lý

3.2.2.1 Sơ đồ liên kết các giao diện:

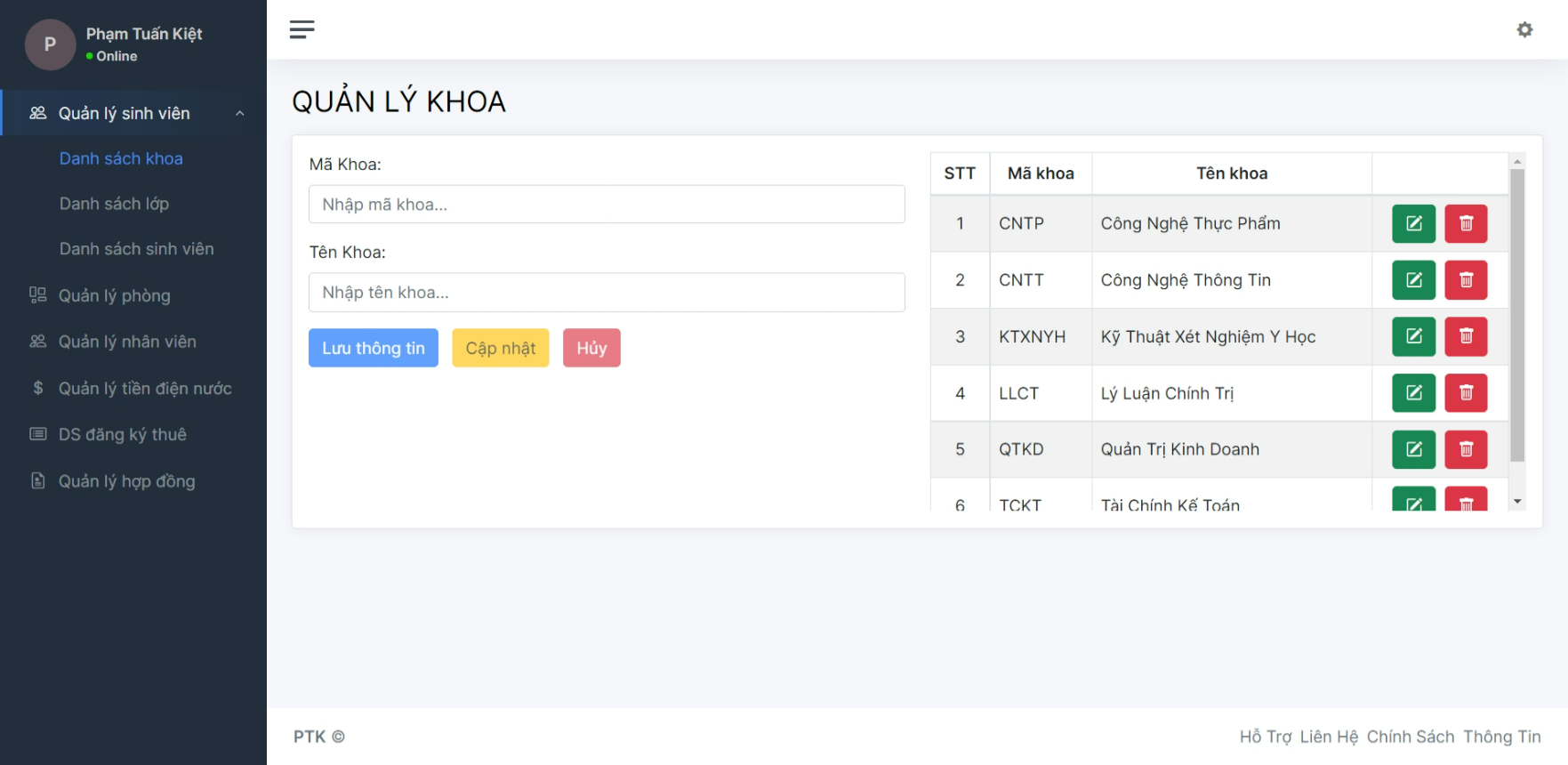


Hình 3.4: Sơ đồ liên kết hệ thống

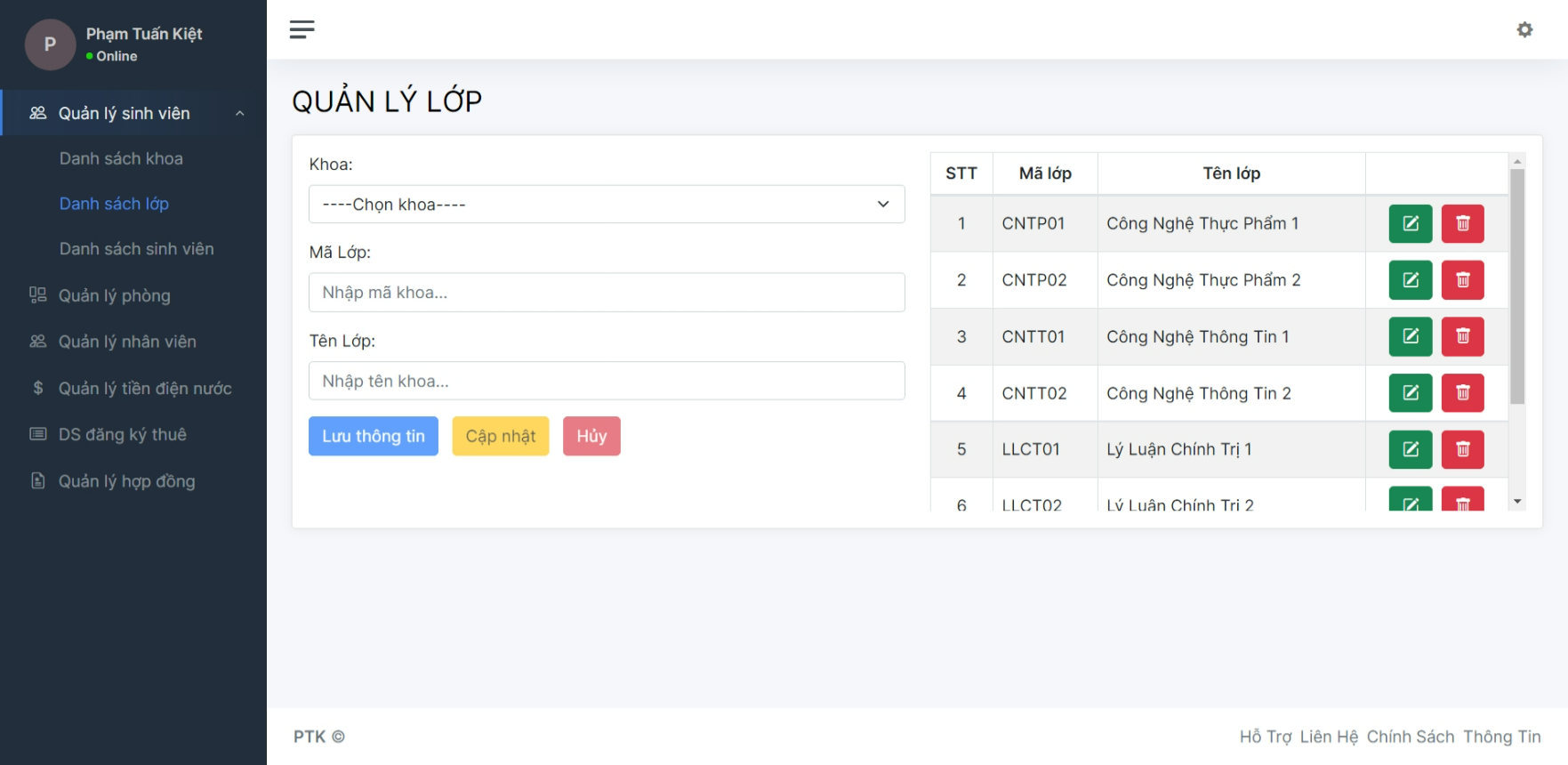
3.2.2.2 Hệ thống các giao diện.



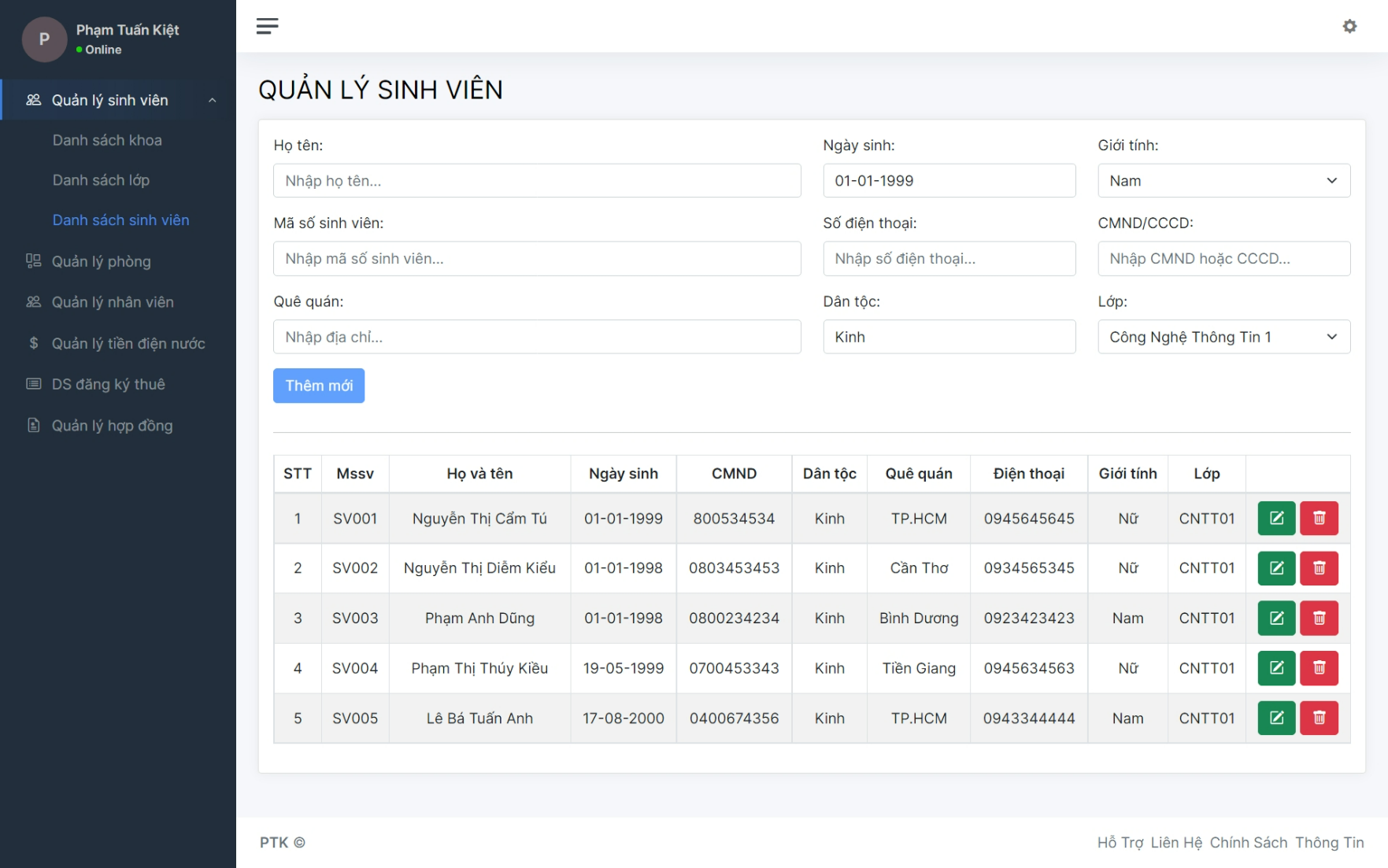
Hình 3.5: Giao diện trang đăng ký thuê phòng



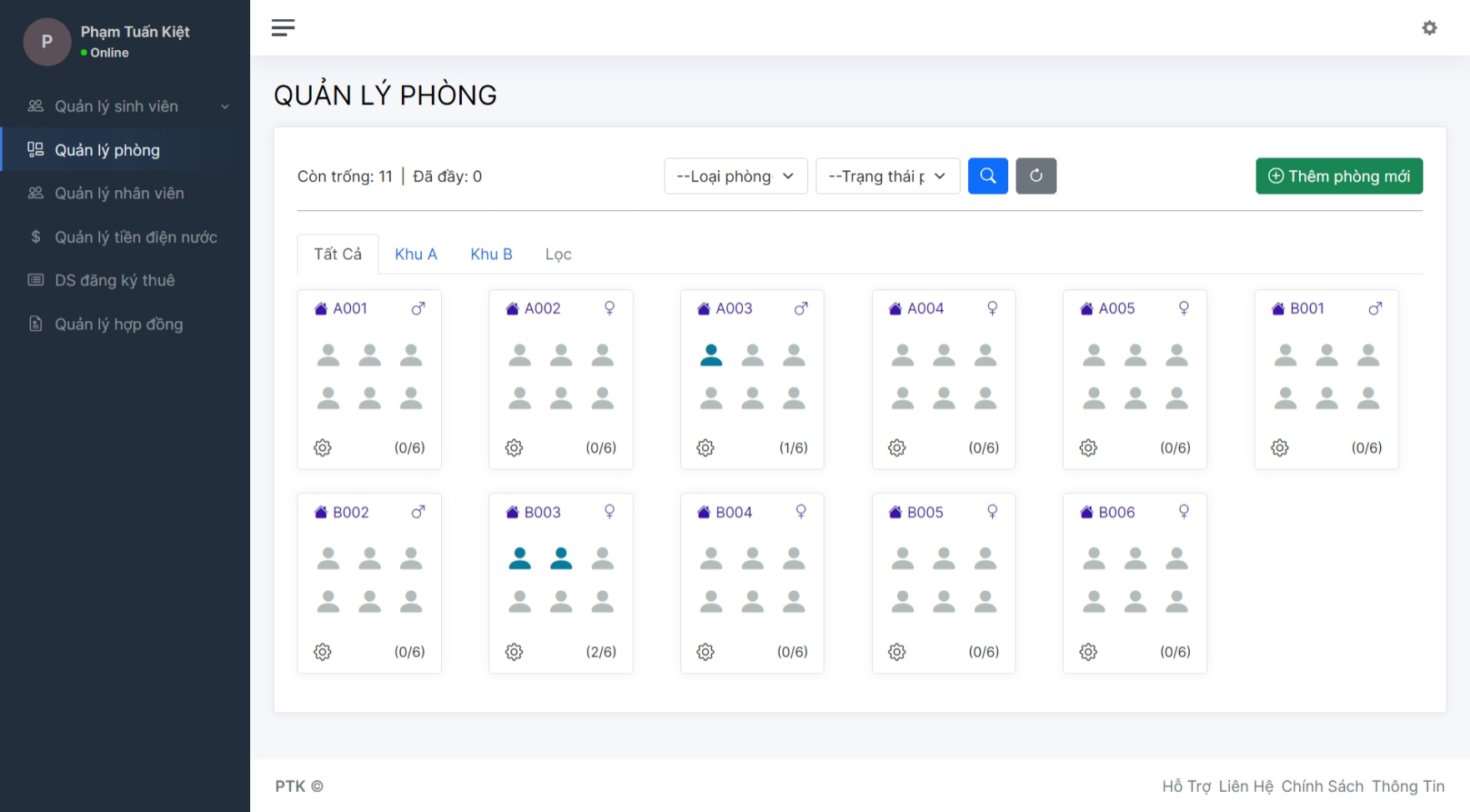
Hình 3.6: Giao diện trang quản lý khoa



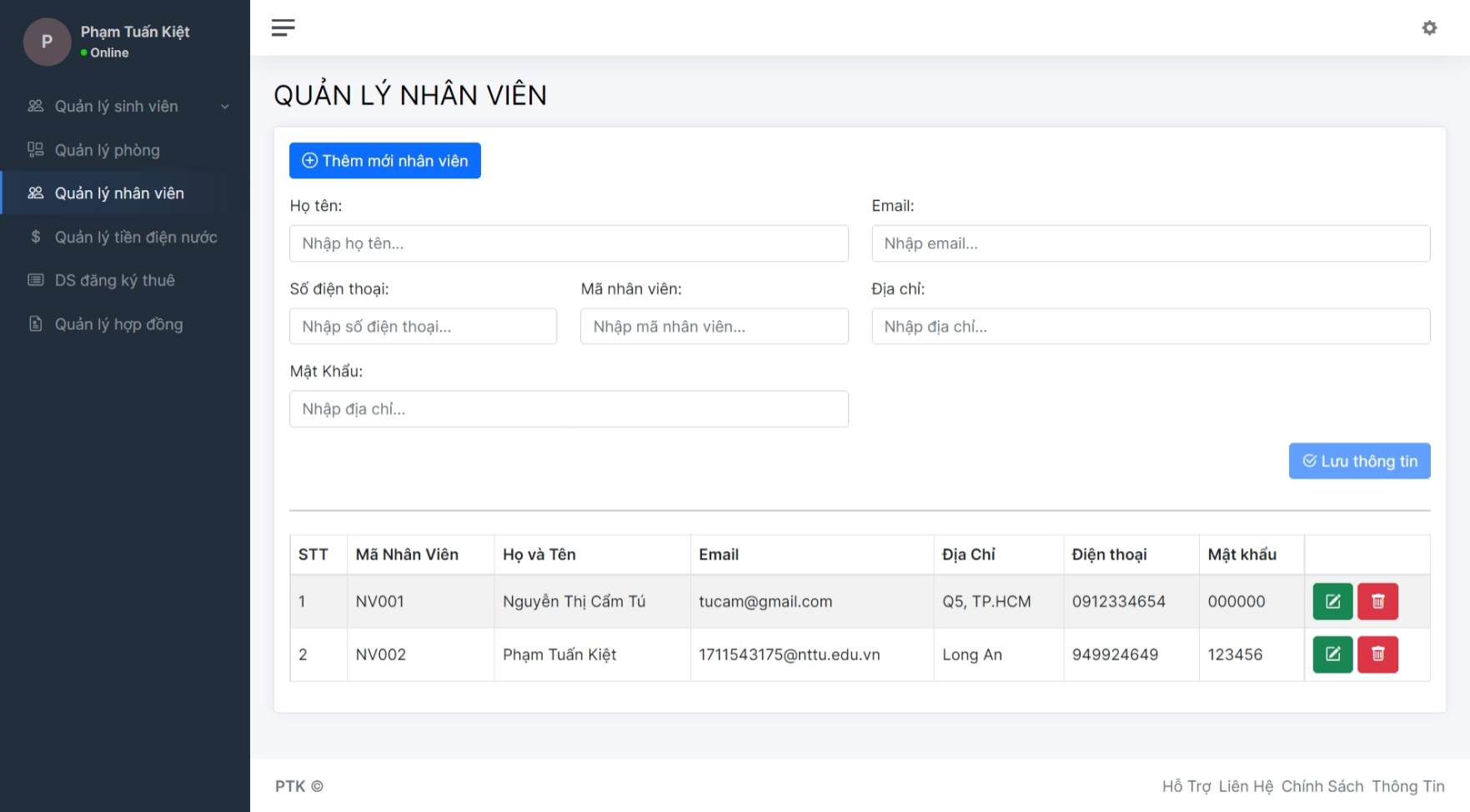
Hình 3.7: Giao diện trang quản lý lớp



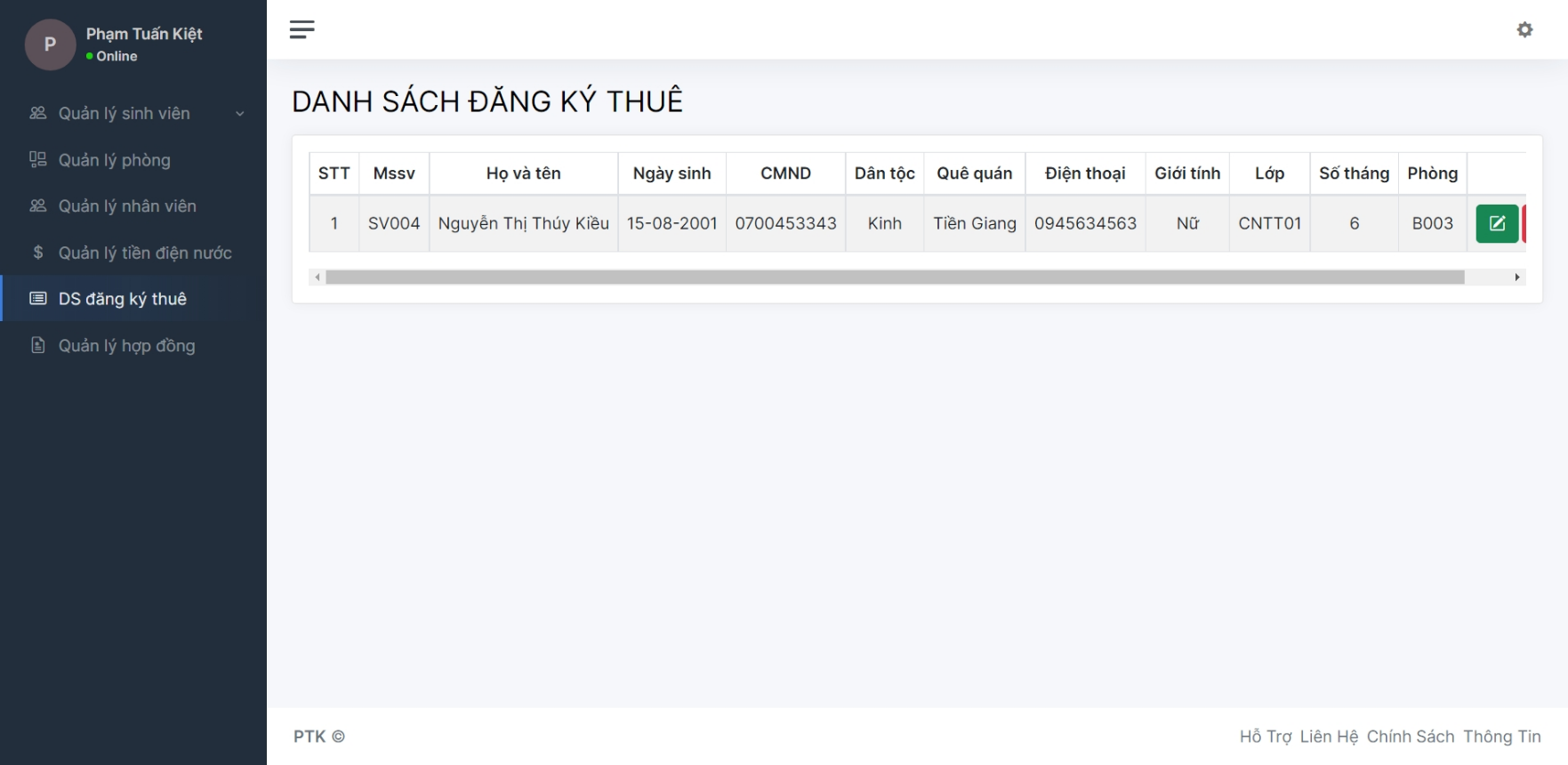
Hình 3.8: Giao diện trang quản lý sinh viên



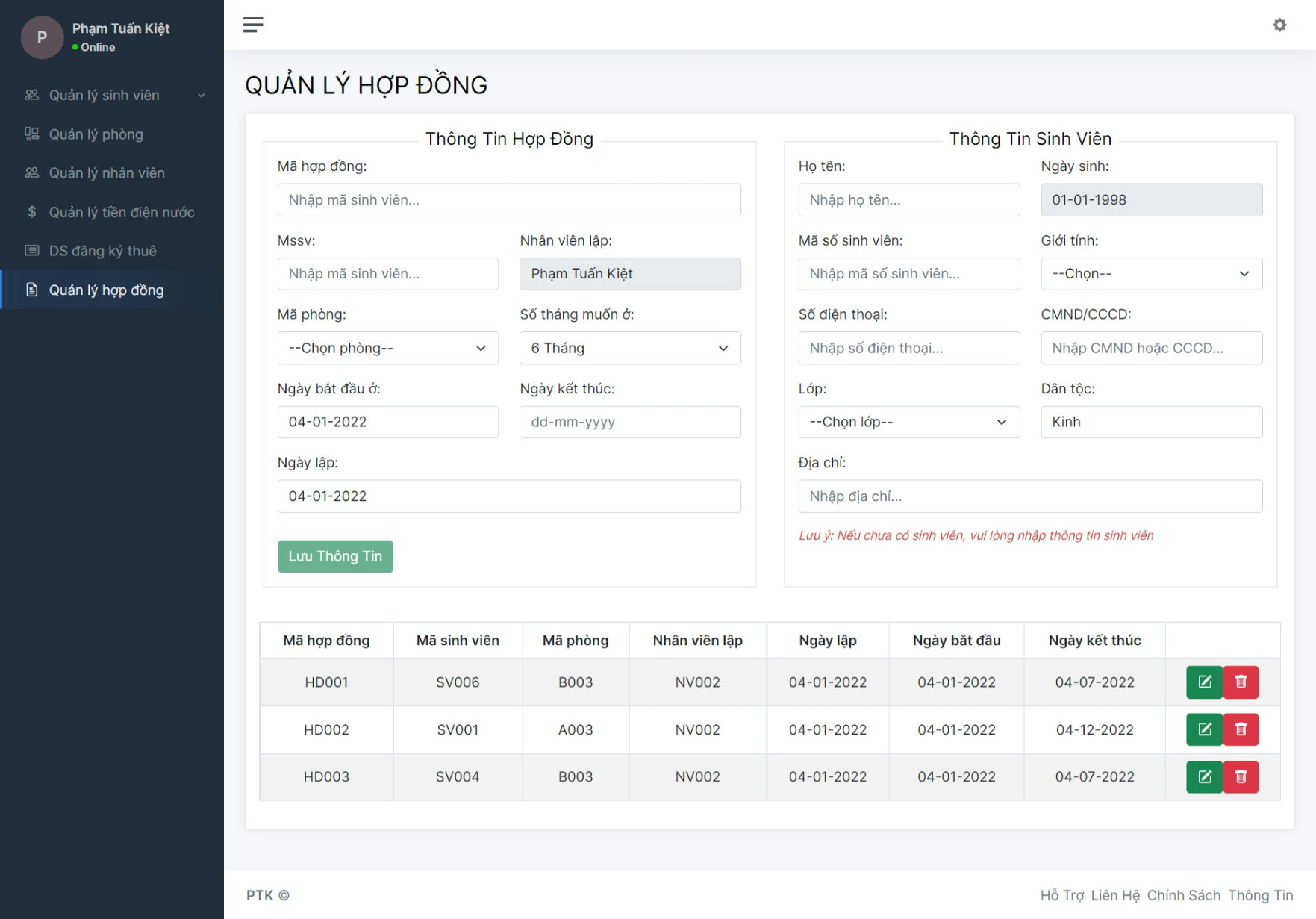
Hình 3.9: Giao diện trang quản lý phòng



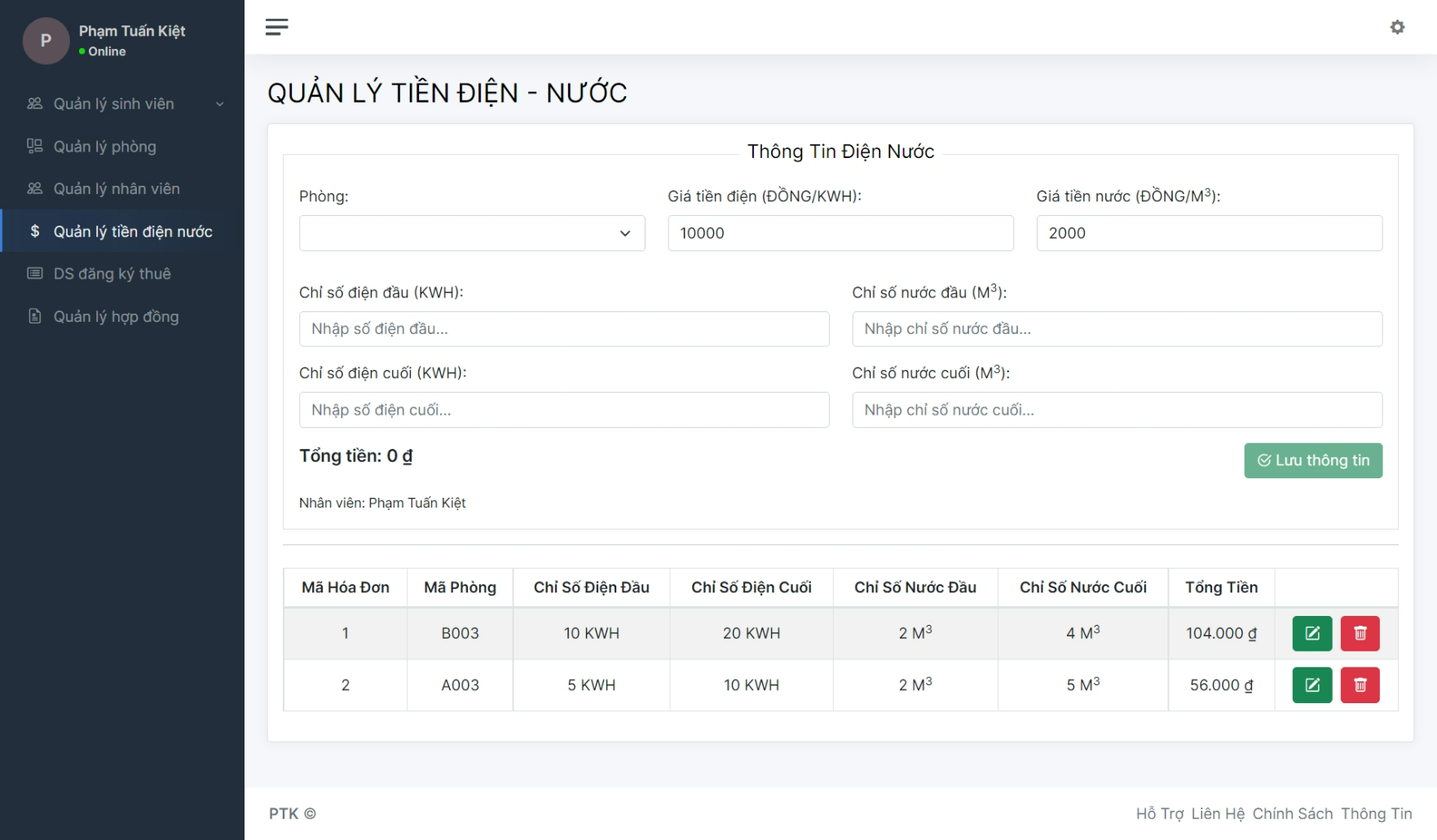
Hình 3.10: Giao diện trang quản lý nhân viên



Hình 3.11: Giao diện trang danh sách đăng ký thuê



Hình 3.12: Giao diện trang quản lý hợp đồng



Hình 3.13: Giao diện trang quản lý tiền điện nước

Link đồ án:

KẾT LUẬN

* **Kết quả đạt được:**

Sau một thời gian thực hiện đề tài, em thực hiện được một số chức năng của trang website như sau:

* Chức năng đăng ký thuê phòng.
* Chức năng tìm kiếm phòng.
* Chức năng đăng nhập.
* Chức năng lập hợp đồng.
* Chức năng lập hóa đơn.
* Các chức năng quản lý như: quản lý sinh viên, quản lý phòng, quản lý hợp đồng, quản lý hóa đơn điện nước và số chức năng quản lý khác.
* **Ưu điểm và nhược điểm:**
* **Ưu điểm:**
* Trang web sử dụng framework Angular giúp việc sử dụng trang web mượt mà hơn không tốn thời gian chờ tải trang.
* Giao diện dễ sử dụng.
* Các chức năng quản lý trực quan dễ dàng sử dụng.
* **Nhược điểm:**
* Chưa có chức năng xuất, in hóa đơn.
* Mặc dù giao diện dễ sử dụng nhưng còn quá đơn giản chưa bắt mắt.
* Chưa có chức năng phân quyền.
* **Hướng phát triển của đồ án:**

Đây là một bài toán có nhiều tiềm năng trong quá trình hội nhập, để phát triển thành một hệ thống hoàn chỉnh và có thể đưa website vào thực tế một cách rộng rãi website cần:

* Cải tiến, hoàn thiện một số chức năng chưa hoàn chỉnh trong hệ thống.
* Xử lý vấn đề bảo mật dữ liệu: Phân quyền, cấp quyền cho từng nhóm người dùng...
* Nâng cấp hệ thống để có thể quản lý thêm những phần khác mà hệ thống hiện tại chưa có.
* Thiết kế giao diện web mang tính chuyên nghiệp hơn.
* Phần mềm ứng dụng được áp dụng cho hầu hết các ký túc xá.
* Phát triển thành một trang web nhằm giúp các chủ ký túc xá yên tâm khi sử dụng trang web với các chức tiện lợi cũng như bảo mật thông và sự nhanh chóng và quản lý công việc của mình trực tiếp trên mạng nhằm phục vụ khách hàng một cách nhanh chóng, thuận lợi hơn.

TÀI LIỆU THAM KHẢO

[1] Phạm Văn Đăng*, Slide bài giảng môn Cơ sở dữ liệu* (2018), Khoa CNTT, Trường ĐH. Nguyễn Tất Thành.

[2] Thái Trúc Nhi*, Slide bài giảng môn Hệ quản trị CSDL* (2018), Khoa CNTT, Trường ĐH. Nguyễn Tất Thành.

[3] Giang Hào Côn, Slide bài giảng môn KTPM 1(2021), Khoa CNTT, Trường ĐH. Nguyễn Tất Thành.

[4] Tài liệu tham khảo về Angular, link: <https://angular.io/docs>, ngày tham khảo 15/12/2021.

[5] Tài liệu tham khảo về Sequelize, link: <https://sequelize.org/>, ngày tham khảo 17/12/2021.

[6] Hướng dẫn sử dụng Sequelize trong Angular:

link: <https://www.bezkoder.com/node-js-express-sequelize-mysql/>, ngày tham khảo 20/12/2021.

link: <https://www.bezkoder.com/angular-12-crud-app/>, ngày tham khảo 21/12/2021.