**TRƯỜNG ĐẠI HỌC ĐÀ LẠT**

**KHOA CÔNG NGHỆ THÔNG TIN**

---🙘🙚---



**BÁO CÁO TIỂU LUẬN: MẪU THIẾT KẾ**

**ĐỀ TÀI: XÂY DỰNG WEBSITE QUẢN LÝ NHÀ TRỌ**

**Giảng viên hướng dẫn: Đỗ Ngọc Cường**

**Sinh viên thực hiện:** 2113005 – Lê Hà Hiếu Nghĩa

2116977 – Trần Hữu Bằng

2115277 – Nguyễn Trung Tín

2111893 – Bùi Phúc Thịnh

**Lớp:** CTK45-PM

**ĐÀ LẠT, 5/2024**

# MỤC LỤC

[MỤC LỤC 2](#_Toc167549529)

[LỜI CẢM ƠN 4](#_Toc167549530)

[BẢNG GIẢI THÍCH THUẬT NGỮ 5](#_Toc167549531)

[1. Lý thuyết 5](#_Toc167549532)

[1.1. Design pattern là gì? 5](#_Toc167549533)

[1.2. Lịch sử phát triển 6](#_Toc167549534)

[1.3. Các Design Pattern hiện nay 7](#_Toc167549535)

[1.3.1. Singleton 7](#_Toc167549536)

[1.3.2. Factory 8](#_Toc167549537)

[1.3.3. Decorator 8](#_Toc167549538)

[1.4. Lợi ích khi áp dụng design pattern 8](#_Toc167549539)

[2. Hệ thống 8](#_Toc167549540)

[2.1. Tóm tắt hệ thống 8](#_Toc167549541)

[2.2. Sơ đồ ngữ cảnh của hệ thống 9](#_Toc167549542)

[2.3. Tóm tắt tính năng của hệ thống 9](#_Toc167549543)

[2.4. Sơ đồ Use Case 10](#_Toc167549544)

[2.5. Danh sách Use Cases 10](#_Toc167549545)

[2.5.1. Đăng nhập 10](#_Toc167549546)

[2.5.2. Đăng thông tin 11](#_Toc167549547)

[2.5.3. Tìm kiếm thông tin 11](#_Toc167549548)

[2.5.4. Cập nhật 11](#_Toc167549549)

[2.6. Ràng buộc và phi chức năng 12](#_Toc167549550)

[2.7. Công nghệ và công cụ sử dụng 12](#_Toc167549551)

[3. Domain Name class 13](#_Toc167549552)

[3.1. Liên kết giữa các thuộc tính 13](#_Toc167549553)

[3.2. Biểu đồ tuần tự 14](#_Toc167549554)

[3.2.1. Đăng nhập 14](#_Toc167549555)

[3.2.2. Tìm kiếm thông tin 14](#_Toc167549556)

[3.2.3. Quản lý người thuê (Cập nhật) 15](#_Toc167549557)

[4. Kết quả 17](#_Toc167549558)

[4.1.1. Mã nguồn 17](#_Toc167549559)

[4.1.2. Giao diện 18](#_Toc167549560)

[5. Kết luận và hướng phát triển 22](#_Toc167549561)

[TÀI LIỆU THAM KHẢO 23](#_Toc167549562)

# LỜI CẢM ƠN

Đầu tiên, chúng em xin chân thành cảm ơn các thầy cô trường Đại học Đà Lạt, Khoa Công Nghệ Thông Tin đã tạo điều kiện cho chúng em có thể tiếp cận với những công nghệ mới cũng như trau dồi thêm kiến thức cho bản thân qua học phần Mẫu thiết kế. Đặc biệt, chúng em xin cảm ơn sự hỗ trợ tận tình cũng như những góp ý chân thành của thầy cô trong quá trình thực hiện đồ án của nhóm chúng em.

Chúng em cũng xin chân thành cảm ơn đến thầy Đỗ Ngọc Cường, giảng viên hướng dẫn đã định hướng, giúp đỡ nhóm trong quá trình hoàn thành đề tài.

Trong khuôn khổ của bài tiểu luận, do thời gian và kinh nghiệm thực tế còn hạn chế nên có những phần thực hiện chưa được tốt, chúng em rất mong nhận được sự góp ý của thầy và các bạn.

Chúng em xin chân thành cảm ơn !

*Đà Lạt, ngày ... tháng 05 năm 2024*

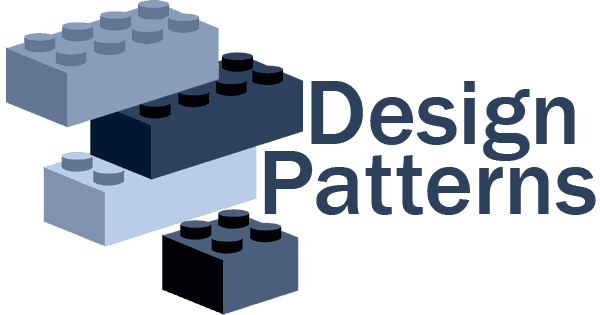
# BẢNG GIẢI THÍCH THUẬT NGỮ

|  |  |
| --- | --- |
| **Từ viết tắt** | **Tên đầy đủ** |
| **CSDL** | Cơ sở dữ liệu |
| **SinhVien** | Đối tượng sinh viên |
| **ChuTro** | Những người cho thuê trọ |
| **QuanTriVien** | Quản trị viên |
| **Tro** | Phòng trọ |

# Lý thuyết

## Design pattern là gì?

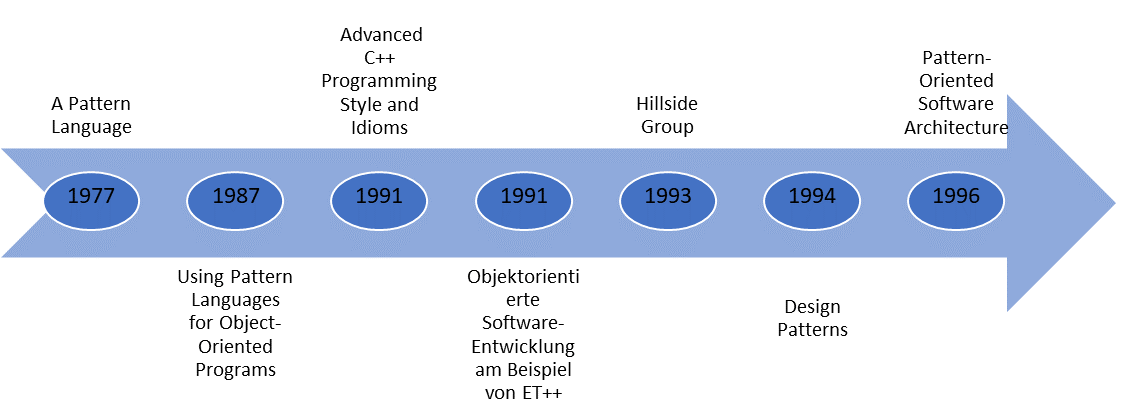
Cần nhấn mạnh rằng: **"*Design Patterns không phải là ngôn ngữ cụ thể nào cả.*"** cho nên chúng ta có thể triển khai trên nhiều ngôn ngữ lập trình khác nhau.



Design pattern là các giải pháp tổng thể được người dùng sử dụng trong thiết kế phần mềm. Những giải pháp này đã được tối ưu hóa, giải quyết trong các tình huống cụ thể. Nó được dùng trong lập trình hướng đối tượng, đây là kỹ thuật lập trình và có thể thực hiện ở nhiều ngôn ngữ như: Java, C#, JavaScript,… Nhờ có Design patterns giúp lập trình viên có được mẫu thiết kế giải quyết các vấn đề chung.

Lập trình viên có thể áp dụng giải pháp này để có thể giải quyết những vấn đề tương tự có  liên quan khác. Các vấn đề mà bạn thường gặp phải thì bạn sẽ có thể tự tìm ra cách để giải quyết nếu như nó chưa được xem là quá tối ưu.

## Lịch sử phát triển



***Hình 1.1: Quá trình phát triển của design pattern trong giai đoạn cuối thế kỷ 20***

Việc sử dụng "pattern" đầu tiên xuất phát từ tác phẩm của kiến ​​trúc sư (cũng là nhà nhân chủng học) Christopher Alexander. Ông viết cuốn sách "***A pattern language***" (1977), trong đó ông trình bày một ngôn ngữ mới, dựa trên những thực thể vượt thời gian được gọi là patterns. Các mẫu mô tả một vấn đề và đưa ra một giải pháp phổ quát [1] [2].

Năm 1987, Ward Cunningham và Kent Beck đã được truyền cảm hứng từ những ý tưởng của Christoph Alexander và áp dụng ý tưởng của ông vào việc phát triển phần mềm. Cùng năm đó, họ xuất bản cuốn sách "***Using Pattern Languages for Object-Oriented Programs***", một ngôn ngữ mẫu để hỗ trợ phát triển giao diện người dùng trong Smalltalk. Họ đã thiết lập hệ thống mẫu đầu tiên để thiết kế giao diện người dùng.

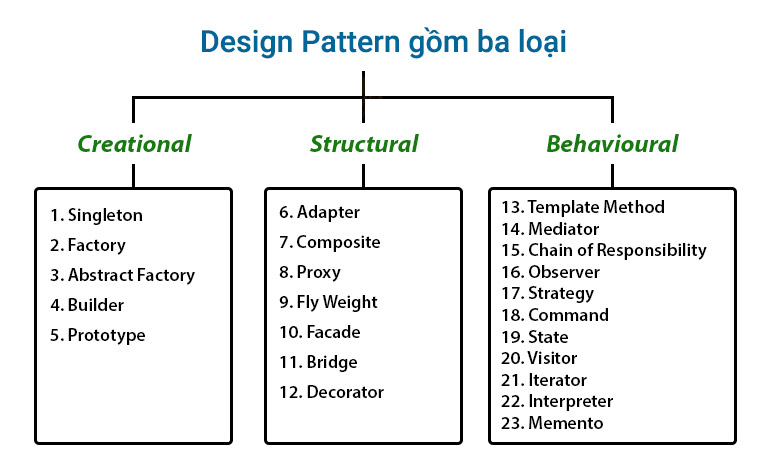
Năm 1991, James Coplien xuất bản cuốn sách " ***Advanced C++ Programming Styles and Idioms*** " [3].Cuốn sách của ông là một danh mục các thuật ngữ trong C++ như:

* Handle/Body: tách biệt giữa giao diện và cách thực hiện
* Envelope/Letter: gắn kết các ngữ nghĩa chung của giao diện và cách triển khai với nhau
* Virtual Constructor: xây dựng hệ thống phân cấp của một kiểu trừu tượng đã biết nhưng thuộc loại cụ thể chưa xác định
* Concrete Data Type: xác định đối tượng sẽ được tạo trên vùng heap của phạm vi hiện tại
* Algebraic Hierarchy: lớp cấu trúc theo các kiểu đại số
* Non-Hierarchical Addition: xử lý các phép tính số học giữa các loại không thể được thăng cấp sang loại khác

Từ năm 1995, các mẫu thiết kế phần mềm hướng đối tượng đã được chứng minh là nguồn tài nguyên quý giá cho cộng đồng công nghệ phần mềm.

## Các Design Pattern hiện nay

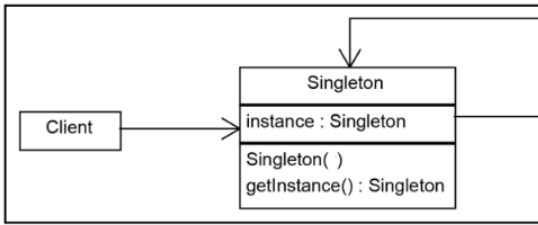
Hệ thống các mẫu Design pattern hiện có 23 mẫu được định nghĩa trong cuốn “***Design patterns:Elements of Reusable Object Oriented Software***” [4] và được chia thành 3 nhóm: Creational, Structural, Behavioral. Chúng được tóm tắt ở hình dưới:



***Hình 1.2: Phân loại design pattern theo mục đích sử dụng***

### Singleton

Khi một lớp được thiết kế theo Singleton sẽ có phương thức trả về duy nhất chính là biểu hiện của nó. Nếu dấu hiệu đó chưa tồn tại thì nó sẽ được tự động tạo mới trước khi trả về. Quá trình này thường được thực hiện bằng cách sử dụng biến tĩnh hoặc biến toàn cục để lưu trữ phương thức tĩnh, hỗ trợ truy cập đến biểu tượng đó. Nó đảm bảo rằng một class chỉ có duy nhất một instance.



***Hình 1.3: Mô tả diagram cho Singleton pattern***

Trong các trường hợp cần quản lý tài nguyên có giới hạn như cơ sở kết nối cơ sở dữ liệu, Singleton góp phần vào việc đảm bảo khả năng sử dụng được dùng để tiết kiệm tài nguyên.

### Factory

Factory Method cung cấp một interface, phương thức trong việc tạo nên một đối tượng (object) trong class. Nhưng để cho class con kế thừa của nó có thể ghi đè để chỉ rõ đối tượng (object) nào sẽ được tạo. Factory method giao việc khởi tao một đối tượng (object) cụ thế cho lớp con (subclass).

Nhiệm vụ của Factory Pattern là quản lý và trả về các đối tượng theo yêu cầu, giúp cho việc khởi tạo đổi tượng một cách linh hoạt hơn. Ngoài ra, đây đích thực là một nhà máy theo đúng như cái tên, và nhà máy này sẽ “sản xuất” các đối tượng theo yêu cầu [5].

### Decorator

Đây là một mẫu thiết kế cấu trúc cho phép bạn đưa thêm các chức năng mới vào đối tượng bằng cách đặt các đối tượng này bên trong các đối tượng bao bọc đặc biệt có chứa các hành vi [6]. Điều này không làm thay đổi hay sửa đổi cấu trúc của nó.

Nói theo cách dễ hiểu nhất, sau một thời gian sử dụng, user mong muốn ngoài việc gửi thông báo trên máy tính còn muốn nhận thông báo qua SMS, Facebook, Slack. Để bổ sung thêm 3 tính năng chúng ta tạo 3 subclass kế thừa từ thư viện cha.

## Lợi ích khi áp dụng design pattern

Một số lợi ích khi phát triển design pattern:

Các mẫu thiết kế đã được xác định và cung cấp cách tiếp cận chuẩn theo ngành để giải quyết vấn đề lặp lại, vì vậy sẽ tiết kiệm được thời gian nếu chúng ta sử dụng mẫu thiết kế hợp lý. Ngoài ra các mẫu thiết kế giúp mã dễ hiểu hơn đối với những lập trình viên khác, do đó dễ bảo trì hơn.

Sử dụng các mẫu thiết kế thúc đẩy khả năng tái sử dụng dẫn đến phát triển phần mềm nhanh hơn. Nó giúp giảm tổng chi phí của sản phẩm phần mềm, giúp tăng tốc độ phát triển và giảm thiểu lỗi.

Tăng cường bảo mật: Ví dụ, sử dụng Singleton để quản lý kết nối cơ sở dữ liệu giúp tăng cường kiểm soát và bảo mật dữ liệu. Trải nghiệm người dùng tốt hơn: Việc sử dụng các mẫu thiết kế giúp tối ưu hóa hiệu suất và tính tương tác của trang web, từ đó nâng cao trải nghiệm người dùng.

# Hệ thống

## Tóm tắt hệ thống

Hệ thống nhà trọ Đà Lạt là một trang web tìm nhà trọ, các căn hộ hoặc nhà nguyên căn nhằm đáp ứng cho các sinh viên cũng như những người lao động từ những tỉnh thành khác trên cả nước khi đến để làm việc và học tập. Trang web giúp người dùng tiết kiệm được thời gian cũng như kinh phí sau khi sử dụng trang web.

## Sơ đồ ngữ cảnh của hệ thống

A diagram of a diagram

Description automatically generated

***Hình 1.1 – Mô tả ngữ cảnh của hệ thống***

## Tóm tắt tính năng của hệ thống

|  |  |
| --- | --- |
| **ID** | **Mô tả tính năng** |
| **FEA-01** | **Đăng nhập** |
| **FEA-02** | **Đăng ký** |
| **FEA-03** | **Đăng thông tin liên quan** |
| **FEA-04** | **Tìm kiếm thông tin** |
| **FEA-05** | **Cập nhật thông tin của SinhVien, do ChuTro thực hiện** |

***Bảng 1.1: Tóm tắt tính năng của trang web***

## Sơ đồ Use Case

**A diagram of a network

Description automatically generated**

***Hình 1.2 – Sơ đồ Use Case***

## Danh sách Use Cases

### Đăng nhập

|  |  |
| --- | --- |
| **Use Case UC1: Đăng nhập** | |
| **Phạm vi truy cập** | Tất cả các actor |
| **Luồng sự kiện chính** | 1. Hệ thống hiển thị màn hình đăng nhập 2. Người dùng nhập username, password 3. Nếu thành công, hệ thống thể hiện việc đăng nhập đã hoàn tất 4. Kết thúc use case |
| **Luồng sự kiện phụ** | **1a** Mật khẩu không hợp lệ (khi nhập sai username hoặc password   * Bắt nhập lại thông tin, đi kèm đó là thông báo nhập sai và yêu cầu nhập lại * Quay lại bước 2 trong luồng sự kiện chính   **1b.** Quên mật khẩu: khi người dùng chọn chức năng quên mật khẩu trên màn đăng nhập   * Hiển thị màn hình |

***Bảng 1.2: Mô tả thao tác cho UC đăng nhập***

### Đăng thông tin

|  |  |
| --- | --- |
| **Use Case UC2: Đăng thông tin** | |
| **Phạm vi truy cập** | ChuTro |
| **Luồng sự kiện chính** | 1a: Tạo mới   1. Thực hiện việc đăng nhập 2. Đưa ra form chứa các textbox liên quan, người chủ chỉ cần nhập thông tin liên quan vào 3. Nhấn xác nhận 4. Gửi thông báo thành công, thông tin mới đó được hiển thị lên form 5. Kết thúc thao tác (có thể load lại trang) |
| **Luồng sự kiện phụ** | * Nếu chưa đăng nhập thì phải thực hiện lại bước **1** trong luồng sự kiện chính |

***Bảng 1.3: Mô tả thao tác cho UC đăng nhập***

### Tìm kiếm thông tin

|  |  |
| --- | --- |
| **Use Case UC3: Tìm kiếm thông tin** | |
| **Phạm vi truy cập** | SinhVien |
| **Luồng sự kiện chính** | 1a:   1. Truy cập trang web 2. Thực hiện tìm trên trang chính với ba yếu tố chính: khoảng cách, giá, vị trí 3. Dựa trên thông tin được để lại, có thể thực hiện liên lạc với chủ sở hữu 4. Kết thúc thao tác |
| **Luồng sự kiện phụ** | * Có thể không cần đăng nhập, thực hiện thao tác ngay trên trang chính |

***Bảng 1.4: Mô tả thao tác cho UC đăng nhập***

### Cập nhật

|  |  |
| --- | --- |
| **Use Case UC4: Cập nhật( Thêm xóa, sửa)** | |
| **Phạm vi truy cập** | ChuTro |
| **Điều kiện kiên quyết** | * Phải hoàn thành việc đăng nhập vào web |
| **Luồng sự kiện chính** | 1a: Chức năng thêm   1. Hệ thống hiển thị màn hình quản lý (điền thông tin của người mới đến thuê) 2. ChuTro nhập thông tin liên quan 3. Nếu thành công, hiển thị thêm thông tin thành công, cập nhật trong db 4. Kết thúc thao tác   1b: Chức năng chỉnh sửa hoặc xóa   1. Hệ thống hiển thị màn hình quản lý 2. Nhấn vào nút edit hoặc delete ở góc bên phải 3. Hiển thị thông tin để user có thể thực hiện thao tác 4. Hệ thống gửi form xác nhận có muốn thay đổi hay không? 5. Nếu thành công, hiển thị thông tin đã chỉnh sửa hoặc xóa thành công 6. Kết thúc thao tác |

***Bảng 1.5: Mô tả thao tác cho UC đăng nhập***

## Ràng buộc và phi chức năng

|  |  |  |
| --- | --- | --- |
| **ID** | **Loại** | **Mô tả** |
| NFA-01 | Hiệu suất | Truy cập trang web nhanh |
| NFA-02 | Khả năng sử dụng | Các đối tượng dùng web |
| NFA-03 | Độ tin cậy | Thông tin chính xác |

***Bảng 1.6: Các ràng buộc của web***

## Công nghệ và công cụ sử dụng

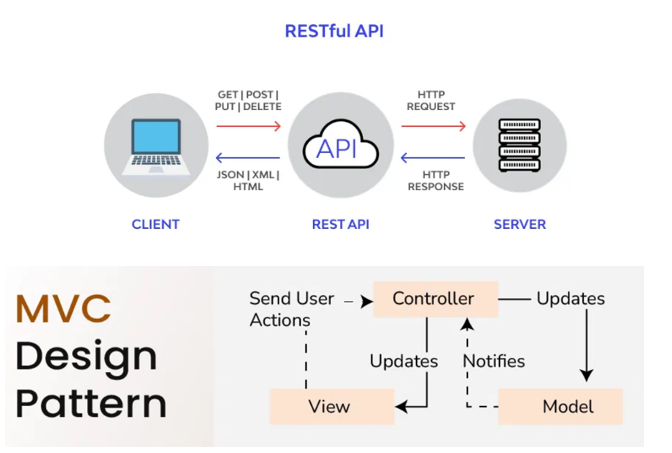
Dự án được viết bằng ngôn ngữ ReactJS, sử dụng CSDL MongoDB

Môi trường phát triển: Visual Studio Code



Các Design Pattern sẽ được áp dụng trong dự án: Rest API và Back-end MVC. Dưới đây là một số lý do:

REST API là một mẫu thiết kế phổ biến cho các dịch vụ web. REST API dựa trên các nguyên tắc kiến trúc nhằm thúc đẩy tính đơn giản, khả năng mở rộng và khả năng tương tác. Nó thường hoạt động chủ yếu dựa vào những giao thức HTTP.



***Hình 1.3: Cách thức hoạt động của backend MVC và RestAPI***

Backend MVC là một mẫu thiết kế phổ biến trong phát triển backend hiện đại. Nó giúp tách biệt lớp trình bày (View) khỏi lớp xử lý logic kinh doanh và lớp lưu trữ dữ liệu (Model và Controller). Sự tách biệt này cho phép các nhà phát triển viết mã mô-đun hóa và dễ bảo trì hơn.

Bên cạnh đó, tính năng xử lý yêu cầu HTTP của Express.js cung cấp một cách mạnh mẽ để xử lý các loại yêu cầu HTTP như GET, POST, PUT, DELETE và các phương thức khác cũng như làm cho HTML cơ bản trong trang web trở nên động hơn.

# Domain Name class

## Liên kết giữa các thuộc tính

|  |
| --- |
| **Mô tả các quan hệ** |
| Một **SinhVien** có thể đến thuê tại một phòng trong một khoảng thời gian nhất định. |
| Một **chủ trọ** có thể cho nhiều **SinhVien** thuê tại một khu trọ |
| Một **chủ trọ** có thể sở hữu nhiều khu trọ (nếu có thể) |
| Một **ChuTro** chỉ có thể có được duy nhất một **DSNguoiThue** |
| Một **Tro** có thể được nhiều người sử dụng trong nhiều lần, và chỉ được sở hữu bởi duy nhất một **ChuTro** |

***Bảng 1.7: Bảng mô tả liên kết các thuộc tính***

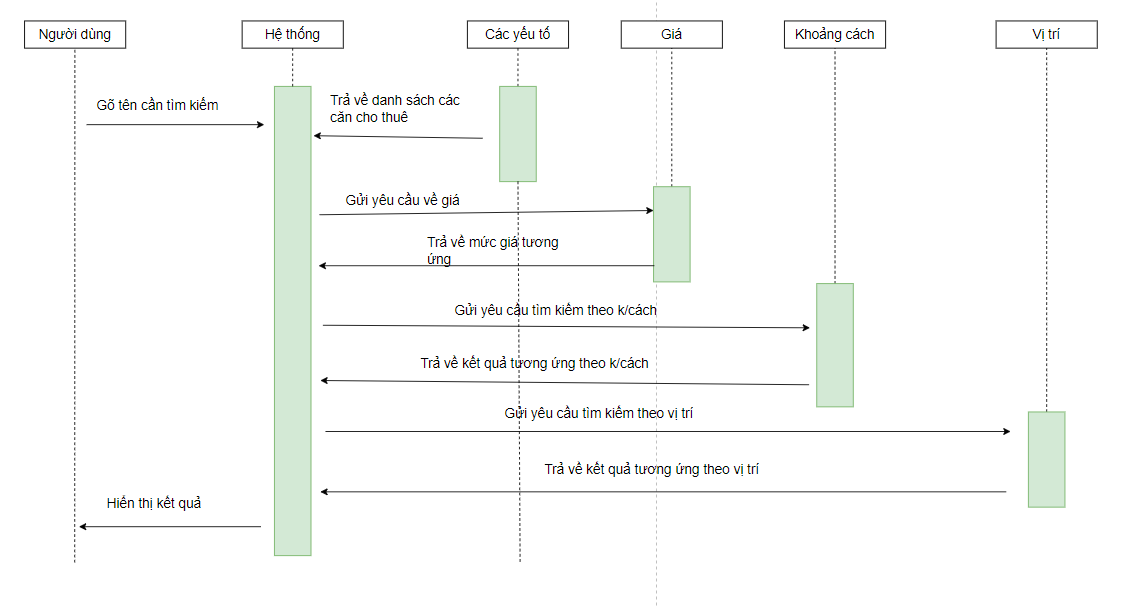
## Biểu đồ tuần tự

### Đăng nhập



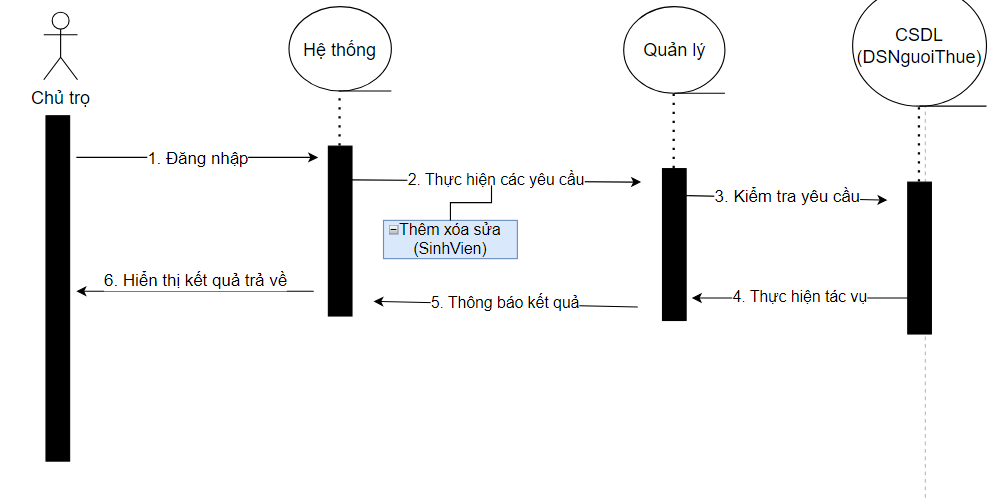
***Hình 1.4: Sơ đồ tuần tự với thao tác đăng nhập***

### Tìm kiếm thông tin

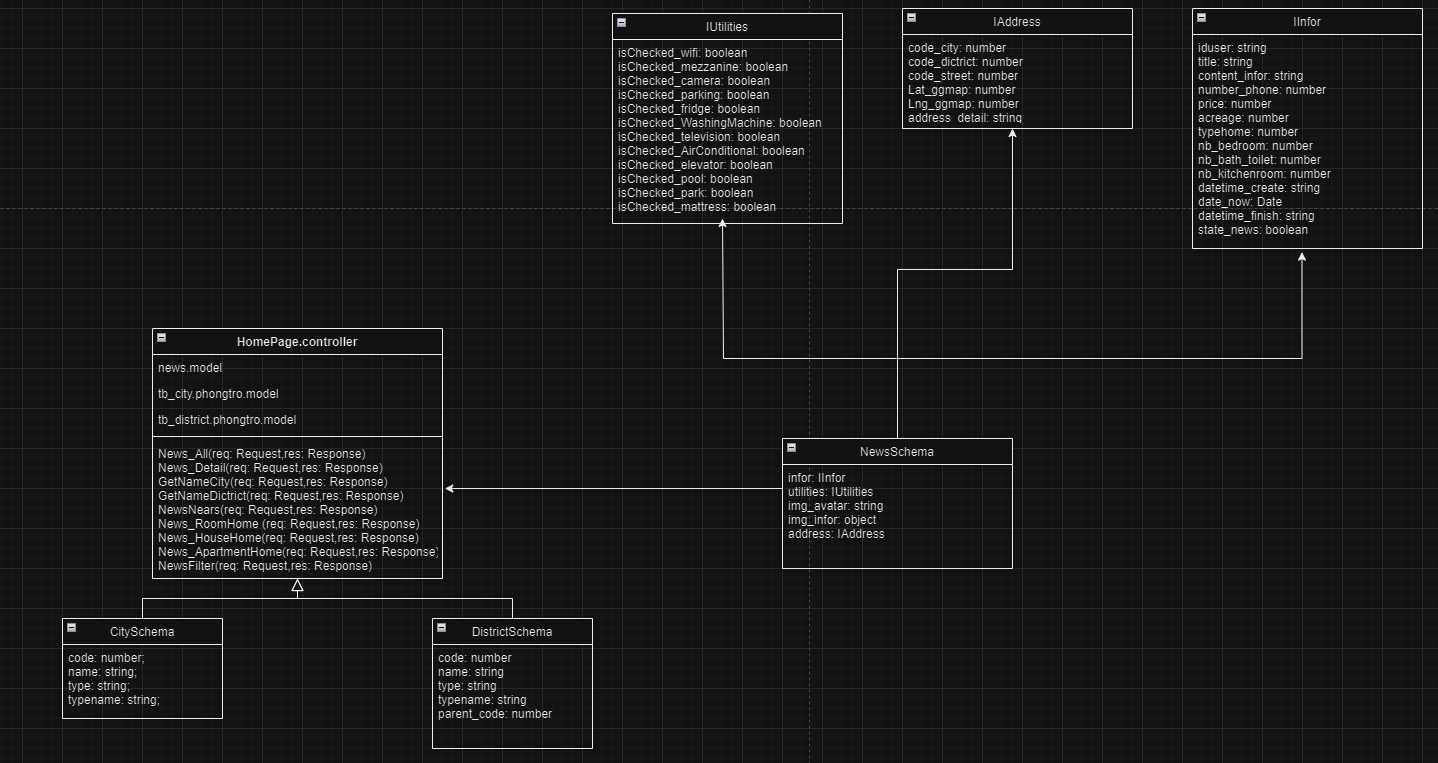
**

***Hình 1.5: Trình tự hoạt động đối với thao tác tìm kiếm thông tin***

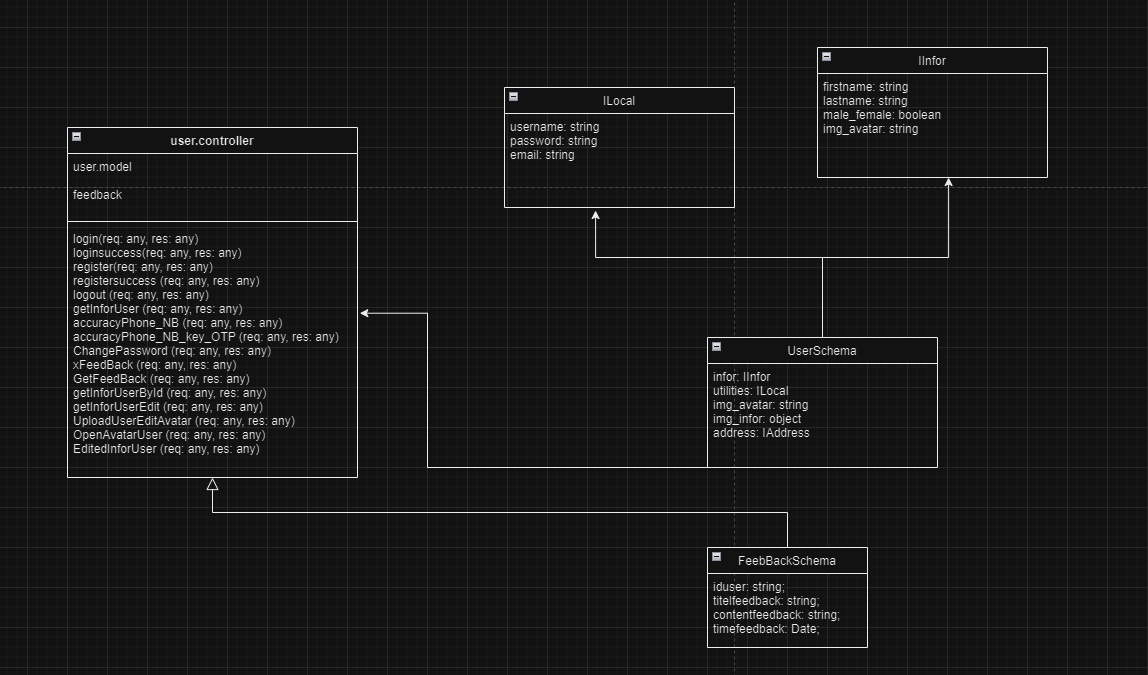
### Quản lý người thuê (Cập nhật)

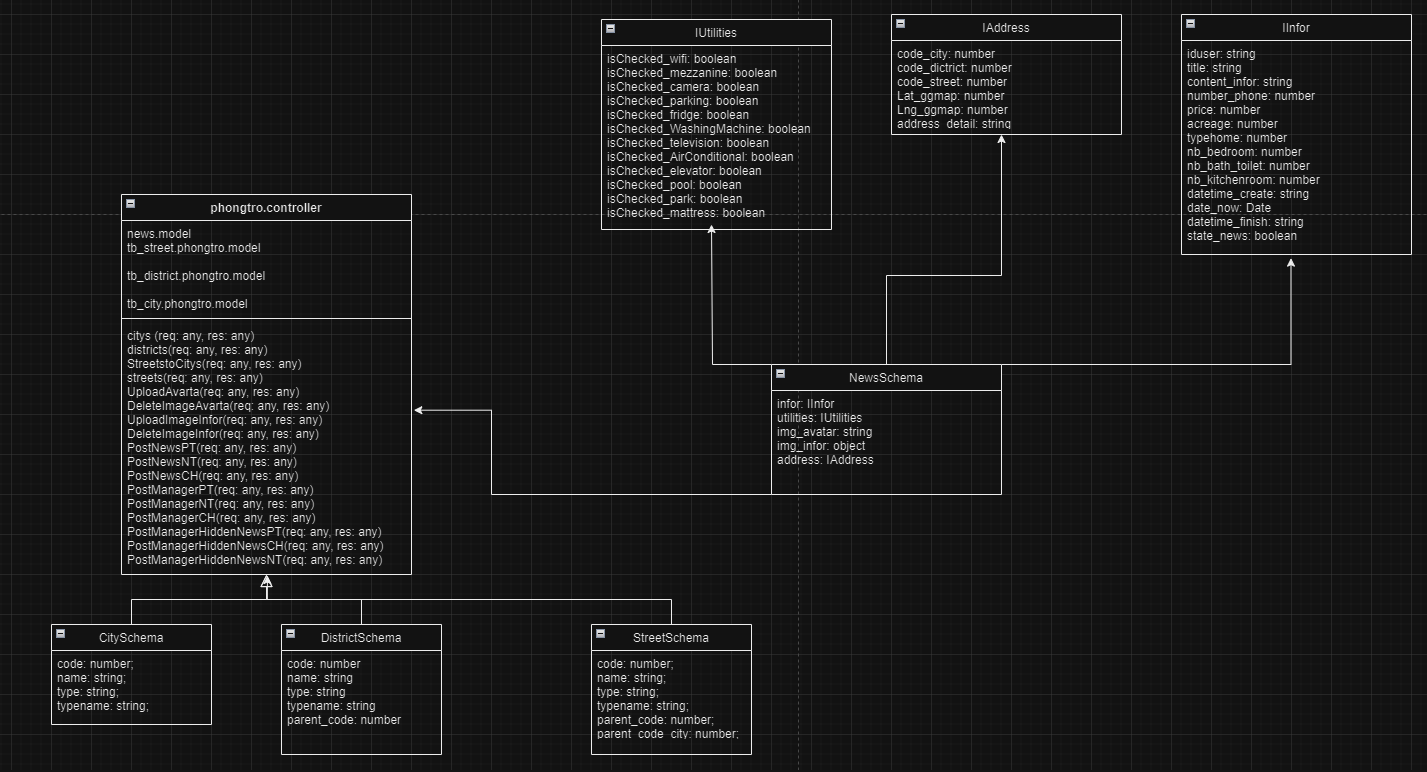
**

***Hình 1.6: Biểu đồ tuần tự cho chức năng quản lý thông tin( Cập nhật)***

* 1. **Biểu đồ lớp**

***Hình 1.7: Biểu đồ lớp cho HomePageCotroller***



***Hình 1.8: Biểu đồ lớp cho UserCotroller***

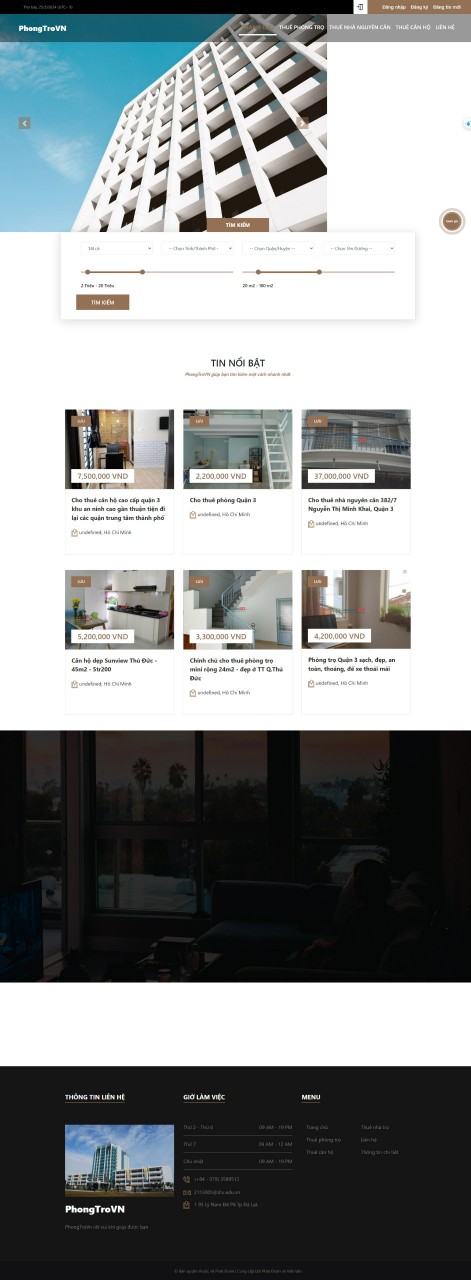
***Hình 1.9: Biểu đồ lớp cho PhongTroCotroller***

# Kết quả

### Mã nguồn

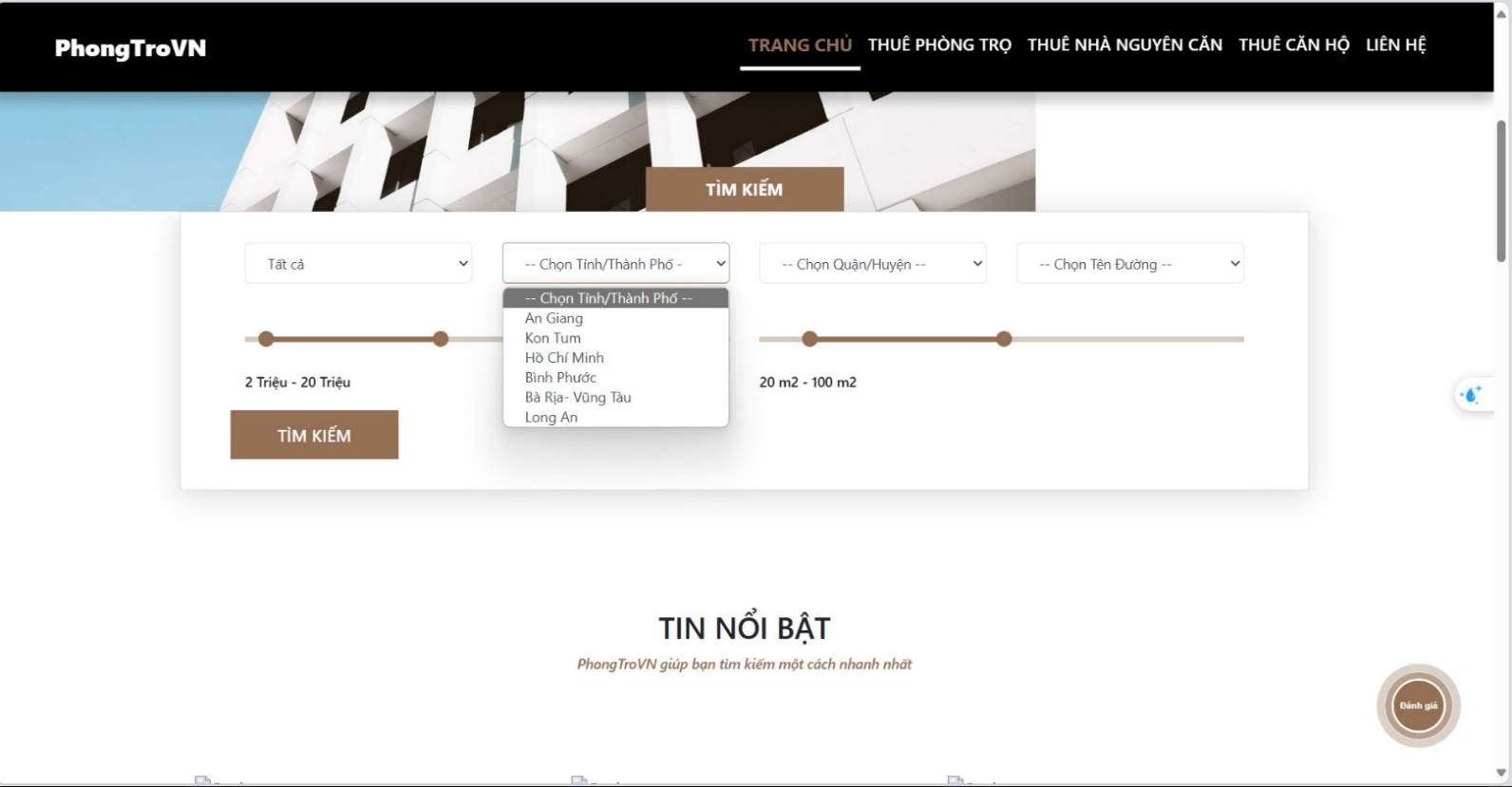
Mã nguồn: <https://github.com/LeHaHieuNghia/QL_NhaTro_VN.git> (đã tích hợp thêm hướng dẫn chạy project)

### Giao diện

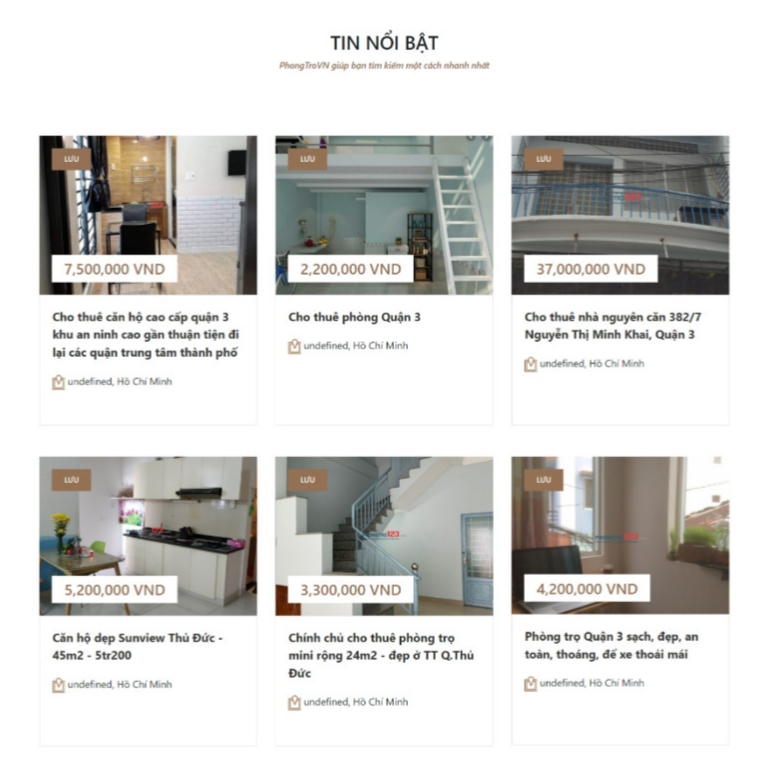


***Hình 1.7: Giao diện chính***

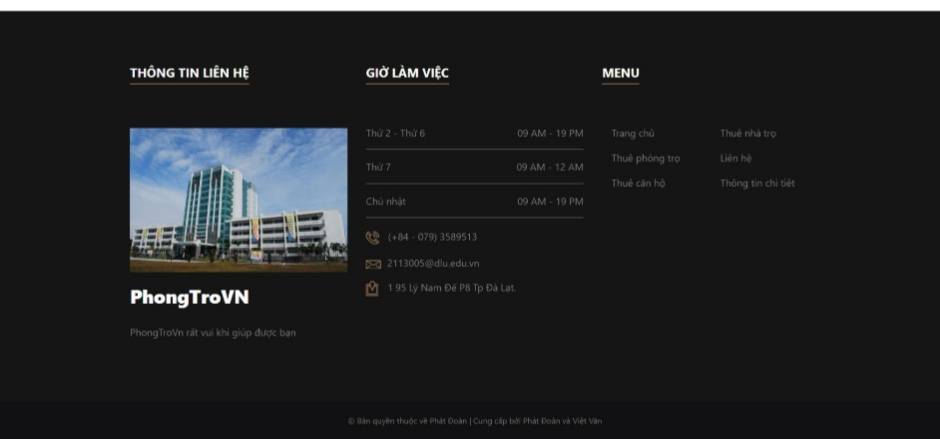
Bao gồm các thanh điều hướng giúp người dùng chuyển sang các hình thức (phòng trọ, nhà nguyên căn, căn hộ). Phía header của trang web sẽ có ba button thực hiện việc đăng nhập đăng ký, truy cập đến các task khác.



Đây là nơi cho phép tìm kiếm thông tin ngay ở UC2. Bao gồm việc tìm kiếm trên ba yếu tố chính:vị trí, giá, khoảng cách.



Tác vụ tin nổi bật và tin phòng trọ cho phép người dùng có thể thấy ngay những thông tin liên quan, giúp việc tra cứu trở nên dễ dàng hơn.



Thanh Footer của trang web

# Kết luận và hướng phát triển

Qua thời gian học tập và nghiên cứu, nhóm đã nâng cao kỹ năng làm việc nhóm, xây dựng thành công một ứng dụng web dựa trên mẫu thiết kế với giao diện đẹp mắt, dễ tương tác, đáp ứng được một số yêu cầu.

Tuy nhiên, do thời gian có hạn và nhóm vẫn chưa lĩnh hội đủ kiến thức kỹ năng để phát triển mạnh hơn nữa những khả năng mà trang web có thể đem đến cho người dùng. Vì vậy trong thời gian tới nhóm cần bổ sung, phát triển thêm các chức năng mới, khắc phục những lỗi hiện tại có thể phát sinh và ứng dụng nhiều pattern hơn cho hệ thống.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Sablier, “Origin of design patterns - Coding Game,” Coding Game, 21 7 2023. [Trực tuyến]. Available: https://www.codingame.com/playgrounds/503/design-patterns/origin-of-design-patterns#:~:text=The%20first%20use%20of%20%22patterns,and%20offer%20an%20universal%20solution.. |
| [2] | S. I. M. S. M. J. I. F.-K. S. A. Christopher Alexander, A Pattern Language: Towns, Buildings, Construction (Center for Environmental Structure Series), Oxford: Oxford University Press, 1977. |
| [3] | J. O. Coplien, Advanced C++ Programming Styles and Idioms, Addison-Wesley Professional, 1991. |
| [4] | Erich Gamma, Richard Helm, Ralph Johnson và John Vlissides, Design patterns: Elements of Reusable Object Oriented Software, 1st biên tập viên, Zürich: Wiley, 2023. |
| [5] | G. Coder, “Hướng dẫn Java Design Pattern – Factory Method,” 12 9 2018. [Trực tuyến]. Available: https://gpcoder.com/4352-huong-dan-java-design-pattern-factory-method/. |
| [6] | Đ. N. Cường, “Decorator,” 3 2024. [Trực tuyến]. Available: https://mtkctk45.notion.site/Decorator-2654eea3c5e24f828000ef9a89c13b41. |