-----🙞🙜🕮🙞🙜-----

**PHẠM TRỌNG NHÂN**

TÊN ĐỀ TÀI

**Xây dựng hệ thống quản lý nhà trọ**

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Hà Nội, Năm 2023**

**ĐỀ TÀI:**

**Xây dựng hệ thống quản lý nhà trọ**

**Hà Nội, Năm 2023**

**CÁN BỘ HƯỚNG DẪN**

**CÁN BỘ PHẢN BIỆN**

**SINH VIÊN THỰC HIỆN**

**Phạm Trọng Nhân**

Mã sinh viên: 1521050085

Số điện thoại: 0775331777

Email: ptnhan109@gmail.com

Lớp: Công nghệ Phần mềm K60

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**Ngành Công nghệ thông tin**

**Chuyên ngành Công nghệ phần mềm**

**Mã số:**

PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

**1. Thông tin chung**

|  |  |
| --- | --- |
| Tên đề tài | Xây dựng hệ thống quản lý nhà trọ |
| Họ và tên sinh viên: | Phạm Trọng Nhân |
| Điện thoại liên lạc: | 0775331777 |
| Email: | Ptnhan109@gmail.com |
| Lớp: | Công Nghệ Phần Mềm K60 |
| Hệ đào tạo: | Đại học chính quy |
| Đồ án tốt nghiệp được thực hiện tại: | Hà Nội |
| Thời gian làm ĐATN: | 2023 |

**2. Mục tiêu của ĐATN**

Nghiên cứu tổng quan về mô hình kỹ thuật công nghệ ASP.NET API, Angular 13 , ứng dụng để xây dựng Website bán quần áo.

**3. Các nhiệm vụ cụ thể của ĐATN**

- Nghiên cứu các kỹ thuật xây dựng ASP.NET API, Angular 12

- Ứng dụng xây dựng Hệ thống quản lý nhà trọ

**4. Lời cam đoan của sinh viên:**

Tôi – Phạm Trọng Nhân – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của ThS. Hoàng Anh Đức

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 10 tháng 7 năm 2023*  Tác giả ĐATN  Phạm Trọng Nhân |

**5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:**

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 10 tháng 10 năm 2023*  Cán bộ hướng dẫn  ThS. Hoàng Anh Đức |

# TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Hiện tại, hình thức kinh doanh cho thuê trọ, cho thuê kí túc xá không còn quá mới lạ tại các khu đô thị lớn, hay gần các khu công nghiệp lớn nơi tập trung dân cư đông đúc. Bất chấp chi phí đầu tư cho hình thức này là cực lớn nhưng bù lại chi phí duy trì thấp, và nguồn thu ổn định nên việc kinh doanh cho thuê trọ vẫn nở rộ trong những năm gần đây. Hình thức kinh doanh lớn là thế nhưng việc vận hành của việc kinh doanh nhà trọ vẫn còn rất lỏng lẻo về các vấn đề như giá thành, hợp đồng, doanh thu thập chí an toàn PCCC,… kéo theo vô số bất cập cả người cho thuê và người thuê. Vì vậy nên em chọn đề tài Xây dựng hệ thống quản lý nhà trọ cho đồ án tốt nghiệp của mình.

# LỜI CẢM ƠN

Trong thời gian làm đồ án tốt nghiệp, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy hướng dẫn và các thầy cô trong khoa - bộ môn Công nghệ phần mềm.

Em xin gửi lời cảm ơn chân thành đến ThS.Hoàng Anh Đức, giảng viên Bộ môn Công nghệ phần mềm - trường ĐH Mỏ - Địa Chất người đã tận tình hướng dẫn, chỉ bảo em trong suốt quá trình làm khoá luận.

Em cũng xin chân thành cảm ơn các thầy cô giáo trong Bộ môn Công nghệ phần mềm Trường Đại học Mỏ - Địa Chất đã dạy dỗ cho em kiến thức các môn chuyên ngành, giúp em có được cơ sở lý thuyết vững vàng và có nhiều góp ý qua những lần bảo vệ tiến độ để đồ án của em được tốt hơn.

Em xin chân thành cảm ơn !

# MỤC LỤC

[PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP I](#_Toc47820363)

[TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP III](#_Toc47820364)

[LỜI CẢM ƠN IV](#_Toc47820365)

[MỤC LỤC V](#_Toc47820366)

[DANH MỤC CÁC HÌNH VẼ VIII](#_Toc47820367)

[DANH MỤC CÁC BẢNG BIỂU IX](#_Toc47820368)

[DANH MỤC CÁC TỪ VIẾT TẮT X](#_Toc47820369)

[MỞ ĐẦU 1](#_Toc47820370)

[CHƯƠNG 1 : TỔNG QUAN VỀ ASP.NET 2](#_Toc47820371)

[1.1 Tổng quan về ASP.NET MVC 2](#_Toc47820372)

[1.2 Những ưu điểm của ASP NET 3](#_Toc47820373)

[1.3 Quá trình xử lý tệp tin ASPX 4](#_Toc47820374)

[CHƯƠNG 2 : PHÂN TÍCH THIẾT KẾ HỆ THỐNG VÀ XÂY DỰNG CƠ SỞ DỮ LIỆU 5](#_Toc47820375)

[2.1 Biểu đồ use case ( Tổng quát ) 5](#_Toc47820376)

[2.2 Use case quản lý đơn hàng 6](#_Toc47820377)

[2.3 Use case quản lý danh mục 7](#_Toc47820378)

[2.4 Use case quản lý sản phẩm 8](#_Toc47820379)

[2.5 Use case quản lý màu 9](#_Toc47820380)

[2.6 Use case quản lý size 10](#_Toc47820381)

[2.7 Mô tả chi tiết các use case 11](#_Toc47820382)

[2.7.1 Đăng nhập 11](#_Toc47820383)

[2.7.2 Quản lý sản phẩm 11](#_Toc47820384)

[2.7.3 Quản lý hóa đơn 12](#_Toc47820385)

[2.7.4 Quản lý thể loại 12](#_Toc47820386)

[2.7.5 Đặt hàng 13](#_Toc47820387)

[2.7.6 Quản lý giỏ hàng 13](#_Toc47820388)

[2.7.7 Tìm kiếm sản phẩm 14](#_Toc47820389)

[2.7.8 Comment 14](#_Toc47820390)

[2.8 Biểu đồ trình tự (đăng nhập) 16](#_Toc47820391)

[2.9 Biểu đồ trình tự (tìm kiếm) 17](#_Toc47820392)

[2.10 Biểu đồ trình tự (quản lý sản phẩm) 18](#_Toc47820393)

[2.11 Biểu đồ trình tự (đặt hàng) 22](#_Toc47820394)

[CHƯƠNG 3 : XÂY DỰNG CƠ SỞ DỮ LIỆU 25](#_Toc47820395)

[3.1 Xây dựng các Relationship 25](#_Toc47820396)

[3.1.1 Các Relationship 25](#_Toc47820397)

[3.1.2 Biểu đồ Diagram 26](#_Toc47820398)

[3.2 Các table 27](#_Toc47820399)

[3.2.1 dbo.Account 27](#_Toc47820400)

[3.2.2 dbo.Category 27](#_Toc47820401)

[3.2.3 dbo.Color 28](#_Toc47820402)

[3.2.4 dbo.DiscountCode 28](#_Toc47820403)

[3.2.5 dbo.Contact 29](#_Toc47820404)

[3.2.6 dbo.DiscountCode 30](#_Toc47820405)

[3.2.7 dbo.Employee 31](#_Toc47820406)

[3.2.8 dbo.Order 32](#_Toc47820407)

[3.2.9 dbo.OrderDetail 33](#_Toc47820408)

[3.2.10 dbo.OrderStatus 34](#_Toc47820409)

[3.2.11 dbo.PaymentID 34](#_Toc47820410)

[3.2.12 dbo.Product 35](#_Toc47820411)

[3.2.13 dbo.ProductSize 36](#_Toc47820412)

[CHƯƠNG 4 : XÂY DỰNG GIAO DIỆN VÀ CHẠY THỬ NGHIỆM 37](#_Toc47820413)

[4.1 Giới thiệu thư viện Bootstrap 37](#_Toc47820414)

[4.1.1 Định nghĩa 37](#_Toc47820415)

[4.1.2 Tại sao nên sử dụng Bootstrap 37](#_Toc47820416)

[4.2 Một số giao diện đã xây dựng 39](#_Toc47820417)

[4.3 Thiết kế code-behind 45](#_Toc47820418)

[4.3.1 Giới thiệu mô hình 3 lớp 45](#_Toc47820419)

[4.3.2 Định nghĩa 45](#_Toc47820420)

[4.3.3 Ưu điểm 45](#_Toc47820421)

[4.4 Xây dựng Code-behind 45](#_Toc47820422)

[HUMG-Clothes.DTO 45](#_Toc47820423)

[4.4.1 HUMG-Clothes.DAL 46](#_Toc47820424)

[4.4.2 HUMG-Clothes.BLL 46](#_Toc47820425)

[4.4.3 HUMG-Clothes.GUI 47](#_Toc47820426)

[TÀI LIỆU THAM KHẢO 48](#_Toc47820427)

# DANH MỤC CÁC hình vẽ

[Hình 1.1 Cơ chế ASPX 4](#_Toc47603552)

[Hình 1.2 Quá trình xử lý tập tin ASPX 5](#_Toc47603553)

[Hình 2.1 Biểu đồ use case tổng quát 6](#_Toc47603554)

[Hình 2.2 Biểu đồ use case quản lý đơn hàng 7](#_Toc47603555)

[Hình 2.3 Biểu đồ use case quản lý danh mục 8](#_Toc47603556)

[Hình 2.4 Biểu đồ use case quản lý sản phẩm 9](#_Toc47603557)

[Hình 2.5 Biểu đồ use case quản lý màu 10](#_Toc47603558)

[Hình 2.6 Biểu đồ use case quản lý size 11](#_Toc47603559)

[Hình 2.7 Biểu đồ trình tự (đăng nhập) 17](#_Toc47603560)

[Hình 2.8 Biểu đồ trình tự (tìm kiếm) 18](#_Toc47603561)

[Hình 2.9 Biểu đồ trình tự (quản lý sản phẩm) 22](#_Toc47603562)

[Hình 2.10 Biểu đồ trình tự (đặt hàng) 25](#_Toc47603563)

[Hình 2.11 Biểu đồ Diagram 27](#_Toc47603564)

[Hình 3.1 Thư viện Bootstrap 38](#_Toc47603565)

[Hình 3.2 Giao diện Login 39](#_Toc47603566)

[Hình 3.3 Giao diện quản lý đơn hang 40](#_Toc47603567)

[Hình 3.4 Giao diện quản lý sản phẩm 41](#_Toc47603568)

[Hình 3.5 Giao diện thêm mới sản phẩm 42](#_Toc47603569)

[Hình 3.6 Giao diện quản lý danh mục 43](#_Toc47603570)

[Hình 3.7 Giao diện chi tiết đơn hàng 44](#_Toc47603571)

# DANH MỤC CÁC BẢNG BIỂU

[Bảng 2.1 Bảng dữ liệu tài khoản 28](#_Toc47696997)

[Bảng 2.2 Bảng dữ liệu thể loại 28](#_Toc47696998)

[Bảng 2.3 Bảng dữ liệu màu sản phẩm 29](#_Toc47696999)

[Bảng 2.4 Bảng dữ liệu mật khẩu 29](#_Toc47697000)

[Bảng 2.5 Bảng dữ liệu thông tin liên lạc 30](#_Toc47697001)

[Bảng 2.6 Bảng dữ liệu khách hàng 31](#_Toc47697002)

[Bảng 2.7 Bảng dữ liệu mật khẩu 32](#_Toc47697003)

[Bảng 2.8 Bảng dữ liệu nhân viên 33](#_Toc47697004)

[Bảng 2.9 Bảng dữ liệu đặt hàng 34](#_Toc47697005)

[Bảng 2.10 Bảng dữ liệu chi tiết đơn hàng 35](#_Toc47697006)

[Bảng 2.11 Bảng dữ liệu trạng thái đơn hàng 36](#_Toc47697007)

[Bảng 2.12 Bảng dữ liệu thanh toán 36](#_Toc47697008)

[Bảng 2.13 Bảng dữ liệu sản phẩm 37](#_Toc47697009)

[Bảng 2.14 Bảng dữ liệu kích thước sản phẩm 38](#_Toc47697010)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Từ viết tắt | Từ tiếng Anh | Từ và nghĩa tiếng Việt |
| 1 | CSDL | Database | Cơ sở dữ liệu |
| 2 | DB | Database | Cơ sở dữ liệu |
| 3 | DTO | Data Tranfer Object | Đối tượng truyền dữ liệu |
| 4 | BLL | Business Logic Layer | Lớp logic nghiệp vụ |
| 5 | DAL | Data Access Layer | Lớp truy cập dữ liệu |

# MỞ ĐẦU

1. Tổng quan tình hình nghiên cứu thuộc lĩnh vực của đề tài

Hiện tại, hình thức kinh doanh cho thuê trọ, cho thuê kí túc xá không còn quá mới lạ tại các khu đô thị lớn, hay gần các khu công nghiệp lớn nơi tập trung dân cư đông đúc. Bất chấp chi phí đầu tư cho hình thức này là cực lớn nhưng bù lại chi phí duy trì thấp, và nguồn thu ổn định nên việc kinh doanh cho thuê trọ vẫn nở rộ trong những năm gần đây. Hình thức kinh doanh lớn là thế nhưng việc vận hành của việc kinh doanh nhà trọ vẫn còn rất lỏng lẻo về các vấn đề như giá thành, hợp đồng, doanh thu thập chí an toàn PCCC,… kéo theo vô số bất cập cả người cho thuê và người thuê

Cũng chính vì những yếu tố kể trên, nhu cầu xây một hệ thống có thể quản lý nhà trọ là một việc cần thiết

2. Mục tiêu đề tài

Xây dựng Hệ thống quản lý nhà trọ :

* Quán lý được các khu trọ, phòng trọ
* Các dịch vụ của khu trọ và thiết bị trong phòng
* Quản lý doanh thu hàng tháng
* Quản lý hợp đồng & khách hàng
* Báo cáo doanh thu, khách hàng, tình trạng hiện tại của các khu

3. Nội dung thực hiện

* Tìm hiểu về Asp.Net API & Angular
* Tìm hiểu và nghiên cứu nhu cầu thực tế của các chủ trọ trong việc quản lý các khu trọ
* Phân tích và thiết kế hệ thống
* Cài đặt và thử nghiệm hệ thống

4. Bố cục đề tài

* Chương 1 : Tổng quan
* Chương 2 : Phân tích thiết kế hệ thống
* Chương 3 : Xây dựng cơ sở dữ liệu
* Chương 4 :Tổng kết

Kết luận và hướng phát triển

1. : TỔNG QUAN VỀ ASP.NET CORE - angular

## Tổng quan về ASP.NET CORE API

### Tổng quan ASP NET

***1.1.1.1 Nền tảng ASP NET***

ASP.NET là một nền tảng web được sử dụng để tạo các trang web, ứng dụng và dịch vụ web. Đây là sản phẩm tích hợp của các ngôn ngữ HTML, CSS và JavaScript. ASP.NET được ra đời vào năm 2002 và đến nay đã được nâng cấp thành nhiều phiên bản khác nhau.



ASP.NET hoạt động trên HTTP, sử dụng các lệnh và chính sách của HTTP để trình duyệt có thể trở thành giao tiếp song phương của máy chủ.

ASP.NET cung cấp những phát triển để tạo ứng dụng web như:

Web Forms

ASP.NET cung cấp tương tác hướng sự kiện cho các ứng dụng web, nó được sử dụng để phát triển ứng dụng có quyền truy cập dữ liệu. Đồng thời, kiểu phát triển web này cũng sẽ cung cấp dữ liệu cho máy chủ và sự kiện để tạo nên ứng dụng phù hợp. Web Forms được phát triển một cách nhanh chóng nhờ vào sử dụng thư viện điều khiển phong phú đánh dấu HTML.

ASP.NET MVC

Cung cấp các mẫu MVC (Model View Controller) để xây dựng lên trang web động. Những mẫu này được sử dụng cho dữ liệu (Model), giao diện người dùng (View) và logic ứng dụng (Controller). Mô hình này sẽ đảm nhận nhiệm vụ duy trì dữ liệu và logic của ứng dụng. Chế độ xem sẽ là giao diện người dùng và hiển thị dữ liệu. Bộ điều khiển sẽ xử lý yêu cầu của người dùng về phần View và Model này.

ASP.NET web Pages

Được sừ dụng để tạo ra các trang web động kết hợp máy chủ  với HTML một cách nhanh chóng. ASP.NET web Pages sẽ đánh dấu mã của bạn và HTML cùng nhau trong cùng một tệp.

Các thành phần chính của nền tảng ASP.NET cơ bản gồm:

Ngôn ngữ: ASP.NET sử dụng nhiều ngôn ngữ lập trình khác nhau như VB.NET và C#.

Thư viện: ASP.NET có bộ thư viện chuẩn bao gồm các giao diện, các lớp và kiểu giá trị. Bộ thư viện này có thể sử dụng lại cho quá trình phát triển ASP.NET và xây dựng các chức năng của hệ thống.

Thời gian chạy ngôn ngữ chung (CLR): CLR - Common Language Runtime được sử dụng để thực hiện các hoạt động mã. Các hoạt động này sẽ thực hiện xử lý các ngoại lệ và thu gom rác.

***1.1.1.2 Tổng quan ASP NET Core***

ASP.NET Core là một web framework mã nguồn và được tối ưu hóa cho cloud để phát triển các ứng dụng web chạy trên nhiều nền tảng như Windows, Linux và Mac. Hiện tại, nó bao gồm MVC framework được kết hợp các tính năng của MVC và Web API thành một web framework duy nhất.

Các ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên .NET Framework hoàn chỉnh.

Nó đã được thiết kế để cung cấp một framework tối ưu cho các ứng dụng để triển khai tới cloud hoặc chạy on-premises.

Nó bao gồm những modular với các thành phần tối thiểu, do đó bạn giữ được tính linh hoạt trong quá trình xây dựng các giải pháp của mình.

Có thể phát triển và chạy các ứng dụng đa nền tảng từ ASP.NET Core trên Windows, Mac và Linux.

***1.1.1.3 Ưu điểm ASP NET Core***

ASP.NET Core đi kèm với những ưu điểm sau:

1. ASP.NET Core có một số thay đổi kiến trúc dẫn đến modular framework nhỏ hơn.
2. ASP.NET Core không còn dựa trên System.Web.dll. Nó dựa trên một tập hợp nhiều yếu tố của Nuget packages.
3. Điều này cho phép bạn tối ưu ứng dụng của mình chỉ cần những NuGet packages cần thiết.
4. Lợi ích của diện tích bề mặt ứng dụng nhỏ hơn thì bảo mật chặt chẽ hơn, giảm dịch vụ, cải thiện hiệu suất và giảm chi phí.
5. Xây dựng và chạy các ứng dụng ASP.NET Core đa nền tảng trên Windows, Mac và Linux.
6. Công cụ mới giúp đơn giản hóa việc phát triển web hiện đại.
7. Liên kết đơn các web stack như Web UI và API Web.
8. Cấu hình dựa trên môi trường đám mây sẵn có.
9. Được xây dựng dựa trên cho DI (Dependency Injection).
10. Tag Helpers làm cho các Razor makup trở nên tự nhiên hơn với HTML.
11. Có khả năng host trên IIS hoặc self-host.

### Tổng quan về Web API

* + - 1. ***Định nghĩa***

API là viết tắt của Application Programming Interface (Giao diện lập trình ứng dụng), một phần mềm trung gian cho phép hai ứng dụng nói chuyện với nhau. Mỗi khi sử dụng một ứng dụng như Facebook, gửi tin nhắn tức thì hoặc kiểm tra thời tiết trên điện thoại, có nghĩa là bạn đang sử dụng API

Mục đích chính của một API là cung cấp khả năng truy xuất đến một tập các hàm hay dùng, ví dụ, hàm để vẽ các cửa sổ hay các icon trên màn hình. Các API, cũng như hầu hết các interfaces, đều có tính trừu tượng (abstract). Phần mềm muốn cung cấp truy xuất đến chính nó thông qua các API cho sẵn, phải hiện thực API đó. Trong nhiều trường hợp, một API thường là một phần của bộ SDK, hay software development kit. Một bộ SDK có thể bao gồm một API cũng như các công cụ/phần cứng, vì thế hai thuật ngữ này không thay thế cho nhau được.

***1.1.2.2 Tầm quan trọng của WebAPI***

API là khớp nối giữa các thành phần phần mềm.

Giả sử bạn có một tính năng cần cung cấp cho module khác phần mềm khác thì bạn sẽ mở một API để tác giả của module/ phần mềm truy cập vào.

Ví dụ, trên các thiết bị điện toán thì hệ điều hành là phần mềm duy nhất có khả năng truy cập tới các thiết bị phần cứng. Do đó, hệ điều hành sẽ phải cung cấp API để ghi file, đọc file, đọc dữ liệu…. Mỗi ứng dụng khi hoạt động sẽ gọi tới API tương ứng của hệ điều hành.

Khi một phần mềm gọi tới API, phần mềm gọi có thể cung cấp dữ liệu đầu vào và đòi hỏi dữ liệu đầu ra từ API hoặc không, nhưng trong mọi trường hợp, phần mềm gọi để có thể tiếp tục hoạt động thì nó cần phần mềm cung cấp phải thực hiện những gì đã cam kết qua API.

### Dữ liệu Json và XML

***1.1.3.1 Dữ liệu JSON***

Điều đầu tiên khi nhắc đến cái tên Json, chắc hẳn chúng ta cần phải nhớ về định nghĩa của nó. Vậy Json là gì, nó có hữu dụng trong công nghệ thông tin hay không? Theo những giải đáp, đây chính là một từ viết tắt của JavaScript Object Notation. Nó giống như một kiểu định dạng dữ liệu theo một quy định. Mà quy định này tất cả hầu hết các ngôn ngữ lập trình đều có thể đọc và hiểu được. Cũng chính vì điều đó mà Json được coi như một tiêu chuẩn mở rộng để có thể trao đổi giữ những thông tin, dữ liệu trên web.

Để tạo ra một file Json, người dùng cần phải tuân thủ đúng cú pháp. Đây chính là điều mà tất cả những người dùng đều phải tìm hiểu. Cú pháp chính là điều cơ bản thiết yếu để các bạn sử dụng Json. Có hai yếu tố cốt lõi trong mỗi Object chính là Keys và Values.

Keys: Chắc chắn đã không còn xa lạ đối với người dùng. Nó phải là chuỗi các ký tự được bao quanh bởi dấu ngoặc kép.

Values: Chính là một kiểu dữ liệu Json hợp lệ. Nó có thể tồn tại ở các dạng khác nhau như array, object, chuỗi hoặc boolean,… Đôi khi Values còn có thể là số hoặc rỗng.

*Các loại values*

Vì values có liên quan mật thiết đến với Object của Json, chính vì vậy chúng ta cũng cần để tâm đến điều này. Values chứa nhiều loại Json hợp lệ. Chúng tôi đã tìm hiểu và tổng hợp một số dưới đây. Các bạn hãy tham khảo ngay nhé!

Array - Đây chính là bộ sưu tập values có thứ tự nhất định. Trong Json, nó được bao quanh bởi dấu ngoặc vuông. Mỗi value bên trong đều được phân tách bằng dấu phẩy. Một dạng Values Array có thể chứa Object của Json. Nói một cách dễ hiểu, nó có thể sử dụng cùng khái niệm cặp key/value. Values phổ biến nhất trong Json là gì chúng ta có thể khẳng định ngay là Array.

Object - Một Object có thể chứa cả value lẫn key. Người dùng có thể để dấu hai chấm sau mỗi key và dấu phẩy sau mỗi value. Điều này được dùng để phân biệt giữa mỗi Object. Trong Json, object được coi như một value. Nó phải tuân theo quy tắc tương tự như một Object.

Chuỗi - Chuỗi cũng được biết đến khá nhiều trong các values của Json. Nó là một bộ tập hợp gồm các số 0 và nhiều ký tự Unicode. Chuỗi được bao quanh bởi hai dấu ngoặc kép.

Số - Bên cạnh chuỗi thì số cũng có thể coi như một value. Số trong Json có thể là số nguyên hoặc số thực.

Boolean - giá trị True hoặc False

Rỗng - Rỗng thể hiện không có thông tin

* + - 1. ***Dữ liệu XML***

Định nghĩa

Cùng giống như JSON, XML là một ngôn ngữ đánh dấu. XML được viết tắt từ tên eXtensible Markup Language, có nghĩa là ngôn ngữ đánh dấu mở rộng. Ngôn ngữ này được World Wide Web Consortium (W3C) kiến nghị tạo nên để góp phần xây dựng các API Service. XML có khả năng truyền dữ liệu và đọc dịch nhiều loại dữ liệu khác nhau. API sẽ trả kết quả về dạng XML để các hệ thống khác nhau có thể nói chuyện với nhau được.

Nếu mô tả một cách đơn giản hơn thì XML giống như một máy phiên dịch ngôn ngữ giữa các hệ thống với nhau vậy. Ngoài chức năng thông dịch, nó còn giúp đơn giản hóa dữ liệu giữa các platform và hệ thống khác nhau. XML cũng có thể được dùng như một kho lưu trữ các dữ liệu đã trao đổi.

**Đặc điểm**

XML được dùng cho những loại dữ liệu có cấu trúc. XML có cấu tạo trực quan khá giống với HTML, nhưng vẫn khác nhau ở nhiều điểm. Có thể nói, XML là cầu nối để đưa HTML đến với XHTML.

Tuy XML tồn tại như một dạng văn bản nhưng nó không dùng để đọc, nó chỉ giúp các hệ thống khác nhau có thể thấu hiểu lẫn nhau. Khi ta nhắc đến định nghĩa đọc file XML trong Java thì có nghĩa là dùng XML để mã hóa Java theo ngôn ngữ đánh dấu, chứ không phải mở văn bản XML ra rồi đọc nó lên.

Vậy định dạng XML là gì? XML có nhiều cách để định dạng khác nhau, dưới đây là một số định dạng phổ biến mà bạn có thể biết:

RSS và ATOM: Chúng đều dùng để mô tả cách làm thế nào mà ứng dụng đọc xử lý được nguồn cấp web.

Microsoft .NET: Hệ thống này sẽ sử dụng XML cho các file cấu hình của nó.

Các phiên bản kể từ Microsoft Office 2007: Chúng sử dụng XML làm cơ sở cho cấu trúc tài liệu. Ký hiệu “X” trong định dạng tài liệu Word .DOCX cũng xuất phát từ đây.

**Khả năng đọc**

Tuy file XML không được dùng để đọc trực tiếp nhưng nó vẫn vô cùng hữu ích đối với các nhà phân tích dữ liệu. Trước mỗi thông tin quan trọng trong XML đều có tag name đầy đủ với ký tự text thân thiện, phù hợp với khả năng đọc hiểu của con người chứ không chỉ là ngôn ngữ máy tính khô khan. Các thông tin được sắp xếp gọn gàng và có tổ chức nên bạn có thể nhanh chóng tra cứu khi cần thiết. Ngoài ra, ngôn ngữ XML tương thích với hầu hết các loại dữ liệu, tạo nên sự trao đổi nhanh chóng giữa máy tính và file cần xử lý.

**Khả năng tương thích**

Các file XML mang tính độc lập. Điều này có nghĩa là bạn có thể tùy ý di chuyển chúng và sử dụng ở bất cứ vị trí, điều kiện nào. Điều kiện cần thiết để sử dụng XML là phải có phần mềm xử lý. Khi đó bạn có thể lưu trữ và truyền đi dữ liệu của mình.

**Khả năng tùy chỉnh**

XML là một loại ngôn ngữ đánh dấu mở rộng. Vì thế nó cho phép người dùng thoải mái tạo các tag riêng hoặc sử dụng các tag có sẵn từ người khác. Điều kiện để sử dụng tag của người khác cũng rất đơn giản. Bạn chỉ cần đảm bảo có ngôn ngữ tự nhiên của domain và chúng sở hữu đầy đủ các tính năng cần thiết. Số lượng tag được tạo cũng sẽ không bị giới hạn trong XML.

## Tổng quan về Angular Framework

### Định nghĩa

Angular là một mã nguồn mở (open source) hay Javascript framework hoàn toàn miễn phí chuyên dụng dành cho công việc viết giao diện web. Đây là sản phẩm được viết bởi Misko Hevery và Adam Abrons (một người bạn của Misko Hevery). Sau đó, Angular chính thức được phát triển và duy trì bởi Google từ năm 2009.

Hiểu theo cách đơn giản, Angular là khung làm việc của Javascript MVC phía client (máy khách) với mục đích phát triển ứng dụng web động. Angular được xem là framework frontend mạnh mẽ và vô cùng chuyên dụng bởi các lập trình viên thực hiện việc cắt HTML cao cấp.



Angular là một nền tảng phát triển được xây dựng dựa trên Javascript, có khả năng mở rộng quy mô không chỉ các dự án của một nhà phát triển mà còn các ứng dụng cấp doanh nghiệp, nó bao gồm:

Một khuôn khổ xây dựng ứng dụng web có khả năng mở rộng dựa trên các thành phần.

Một bộ sưu tập thư viện tốt sẽ được tích hợp nhiều tính năng như quản lý biểu mẫu, định tuyến, giao tiếp máy khách-máy chủ.

Một bộ công cụ dành riêng cho nhà phát triển để bạn có thể phát triển, xây dựng và cập nhật nhanh chóng các bộ mã.

AngularJS được viết bằng Javascript và là từ dùng để nói đến Angular 1 (ra đời vào năm 2009). Angular 2 ( ra đời năm 2016) trở nên sẽ được gọi chung là Angular và được viết bởi phiên bản nâng cao của Javascript. Angular có sự thay đổi nhiều từ AngularJS, do đó cấu trúc của chúng khác nhau hoàn toàn.

### Lịch sử phát triển

14/09/2016: Angular 2 – Initial Version of Angular

23/03/2017: Angular 4 – Version 4: giảm thiểu code được tạo ra, giảm xuống 60% kích thước tệp được đóng gói, đẩy nhanh quá trình phát triển ứng dụng.

11/11/2017: Angular 5 – Version 5: sử dụng HTTPClient thay HTTP, công cụ build optimizer được tích hợp sẵn vào trong CLI, …

03/05/2018: Angular 6 – Version 6: Cập nhật CLI, command line interface, Multiple Validators, …

18/10/2018: Angular 7 – Version 7: ScrollingModule scroll load dữ liệu, dùng Drag and Drop, cập nhật RxJS 6.3, …

25/08/2019: Angular 8 – Version 8: CLI workflow improvements, Dynamic imports for lazy routes ….

06/02/2020: Angular 9.0: di chuyển tất cả các ứng dụng để sử dụng trình biên dịch Ivy và thời gian chạy theo mặc định, dùng TypeScript 3.6 và 3.7

25/03/2020: Angular 9.1

08/04/2020: Angular 10 – Beta Version

### Các thành phần cơ bản

Thành phần (Components)

Một thành phần Angular là một class Typescript tương tác với một view. Một view bao gồm một thành phần và một template. Một template là HTML kết hợp vơi cú pháp liên kết dữ liệu của Angular. Một thành phần có các trường (fields) và các phương thức (methods) hỗ trợ cho một view. View sau đó sẽ được cập nhật bằng các đặc tính và phương thức này. Angular tạo và hủy tất cả các thành phần khi người dùng di chuyển trong ứng dụng.

Liên kết dữ liệu (Data binding)

Liên kết dữ liệu là một cải thiện lớn của Angular. Nó cho phép dữ liệu truyền qua giữa các thành phần và template của bạn. Nếu không có liên kết dữ liệu, các lập trình viên Angular sẽ phải viết code để đẩy dữ liệu vào giữa các thành phần và template. Liên kết dữ liệu không giới hạn ở giao tiếp giữa một template cha và thành phần, nó còn có thể chuyển dữ liệu giữa một thành phần cha với một thành phần con.

Modules

Angular có các module được dựng sẵn để hỗ trợ các lập trình viên, các module này được gọi là NgModules. Các NgModule luôn luôn được đánh dấu bằng chú thích @NgModule trong Angular. Một số module thường gặp là Forms Module, RouterModule, HttpClientModule, và module thiết kế tư liệu Angular.  
  
Ý tưởng về module đến từ nguyên tắc trách nhiệm đơn. Trong khi một module có thể làm nhiều việc thì ở cấp độ cao hơn, nó sẽ chỉ tập trung vào một công việc cụ thể như forms hoặc định tuyến. Điều này ngăn việc phình ứng dụng (application bloating) và sắp xếp các ứng dụng thành các khối chức năng ngắn gọn. Lazy loading có thể giúp cải thiện thời gian tải và tốc độ.

Services

Là từ chỉ chung cho bất kỳ đoạn code nào phục vụ một mục đích cụ thể cho ứng dụng. Đây là một khái niệm khá rộng nhưng sẽ có vài chỉ dẫn mà một service nên tuân theo. Một Angular service phải tái sử dụng được và tuân theo nguyên tắc trách nhiệm đơn đó là làm một công việc và phải làm tốt công việc đó. Khi một thành phần làm việc với view thì một service sẽ xử lý tất cả những gì không liên quan đến view đó.  
  
Chẳng hạn như những công việc lặp lại bao gồm logic nghiệp vụ, lấy dữ liệu hay xác thực input, những công việc thiết yếu cho ứng dụng nhưng không liên quan đến thị giác sẽ thuộc về phần service. Bằng cách định nghĩa logic này trong service, bạn có thể sẵn sàng inject nó vào bất cứ đâu trong ứng dụng, cung cấp cho bạn khả năng tái sử dụng vô hạn. Tuy nhiên Angular cũng không bắt buộc bạn phải thực hiện bất kỳ điều gì kể trên. Một service phức tạp không phải là hiếm thấy, nhưng một service tốt sẽ tuân theo các quy tắc trên.

### Ưu điểm của Angular

Liên nền tảng (Cross-platform): Với Angular bạn có thể phát triển các ứng dụng web cấp tiến (Progressive Web Applications – PWA). PWA cũng cung cấp trải nghiệm như ứng dụng (app-like) cho người dùng sử dụng các công nghệ web tân tiến. Tùy theo nhu cầu thì bạn có thể triển khai một ứng dụng gốc cũng như cấp tiến. Bộ công cụ phát triển phần mềm kết hợp được gọi là Ionic có thể đưa ứng dụng của bạn lên app store hoặc triển khai nó lên mobile web như một PWA.Bạn cũng có thể phát triển ứng dụng máy tính bằng Angular.  
   
Tốc độ cao và tối ưu hiệu năng: Angular định nghĩa lại máy ảo JavaScipt hiện đại bằng cách chuyển các template thành code. Bởi vây, code viểt tay của bạn cũng có thể được tận dụng trong framework này. Điều tuyệt vời nhất của Angular đó là bạn có thể render code thành HTML & CSS và có cái nhìn đầu tiên về ứng dụng trên các nền tảng khác như Node.js, .NET, PHP,..v..v..  
  
 Thời gian tải của Angular nhanh hơn bất kỳ front-end framework nào ở thời điểm hiện tại. Angular tải cực kì nhanh với bất kỳ router thành phần mới nào. Hơn nữa, code sẽ tự động được chia ra đối với mỗi khi người dùng load và yêu cầu render.  
   
- Các ứng dụng Angular dành cho mọi người: Angular là một nền tảng front-end kì diệu không chỉ cho phép lập trình các ứng dụng ấn tượng, mà nó còn có thể tạo ra các hoạt ảnh cao cấp để nâng cao trải nghiệm người dùng, Các API của Angular linh hoạt đến mức các lập trình viên có thể phát triển các bố trí giao diện phức tạp cũng như các hoạt ảnh với sử dụng ít code.  
  
 Hơn nữa, với các framework kiểm thử đơn vị thông minh như Jasmine và Karma, bạn có thể sửa lại code bất kỳ lúc nào, Angular có hơn 11 module kiểm thử được tích hợp sẵn để đảm bảo code không dính lỗi.  
  
>>> xem thêm: [Khóa học php](https://niithanoi.edu.vn/lap-trinh-web-php.html)  
   
Các tính năng kỹ thuật  
  
- Kiến trúc MVC: Kiến trúc này đã được đề cập đến trong phần đầu của bài viết. Dữ liệu của ứng dụng sẽ được quản lý bởi Model, View quản lý phần hiển thị dữ liệu. Controller sẽ đóng vai trò là bộ nối giữa lớp View và lớp Model. Thông thường trong kiến trúc MVC, bạn có thể chia nhỏ ứng dụng một cách tùy ý và viết code để nối nó lại với nhau. Tuy nhiên trong Angular, lập trình viên chỉ cần chia nhỏ ứng dụng thành các phần của MVC và framework sẽ tự làm những công việc còn lại. Điều này sẽ tiết kiệm một cách đáng kể thời gian code.  
   
- Liên kết dữ liệu hai chiều: Một trong những tính năng hiệu quả nhất của Angular đó là hệ thống liên kết dữ liệu hai chiều. Lớp View thể hiện chính xác lớp Model và hai lớp này sẽ phối hợp đồng bộ với nhau một cách hoàn hảo. Nếu bạn thay đổi bất kỳ cái gì trong Model, người dùng có thể thấy rõ thay đổi đó trong View.  
   
- Framework ít code: Angular là một framework ít code so với các công nghệ front-end khác. Bạn không cần phải viết code riêng để  kết nối các lớp MVC. Và bạn cũng không cần đến các code cụ thể để xem một cách thủ công. Các chỉ thị được tách biệt với code ứng dụng. Tất cả sẽ tiết kiệm công sức và thời gian cần thiết để phát triển ứng dụng.  
   
- Angular CLI (Giao diện dòng lệnh): Angular CLI tuân theo các nguyên tắc làm việc tổt nhất của lập trình front-end với các tính năng hỗ trợ SASS/SCSS (Chương trình tiền xử lý CSS - CSS preprocessor) và định tuyến. Hơn nữa Angular CLI thông thường như ng-new hay ng-add hỗ trợ các lập trình viên dễ dàng khám phá các tính năng có sẵn. Các phần cơ bản của CLI bao gồm:

ng New: Là lệnh đầu tiên và cơ bản nhất. Bạn cũng có thể tạo workplace mới với lệnh này.

ng Generate: Cho phép bạn tạo các thành phần, service, tuyến, và các đường ống mới. Bạn cũng có thể tạo các test shell đơn giản với Ng Generate.

ng Serve: Cho phép bạn kiểm thử ứng dụng Angular trên một máy chủ cục bộ.

Test, Lint: Hoàn thiện code của bạn với và các phép kiểm thử đơn vị và end-to-end. Và ng lint cũng giúp chạy các chương trình để phân tích code và tìm ra các lỗi tiềm ẩn.

- CDK và Angular Material: Angular nâng cấp Bộ công cụ phát triển thành phần (Component Development Kit – CDK) theo mỗi phiên bản mới. Làm mới (Refreshing) và cuộn ảo (virtual scrolling) là các tính năng tiêu biểu nhất của Angular CDK. Nó cũng hỗ trợ cho việc tải động cũng như dỡ tải DOM để tạo ra một danh sách lớn các dữ liệu hiệu năng cao. Bạn có thể import ScrollingModule hay DragDropModule vào ứng dụng.   
  
 Một số chức năng chính của DragDropModule bao gồm sắp xếp trong một danh sách, kéo thả tự do, xem trước, và các phần giữ chỗ. Thêm nữa, nó còn có thể sắp xếp lại thứ tự danh sách (moveItemInArray) và chuyển các vật thể qua lại giữa các danh sách (transferArrayItem).  
   
- Cuộn ảo: Tính năng cuộn ảo của Angular giúp code phản ứng với các sự kiện cuộn. Ngoài việc tải và dỡ tải các phần tử DOM có thể thấy được, cuộn ảo còn cung cấp chức năng giả lập vật thể.  
   
- TypeScript: Với TypeScript, Angular cung cấp một trải nghiệm liền mạch cho các lập trình viên. Nó rất hiệu quả trong việc phát hiện lỗi. TypeScript cũng tự động gắn các thiết lập của tệp root dể biên dịch dễ dàng hơn. So với JavaScript, TypeScript có các kiểu liệt kê, giao diện, các lớp generic, các kiểu kết hợp, các kiểu union/intersection, các bộ điều chỉnh truy cập đặc sắc hơn.  
   
- Inject các phần phụ thuộc: Chức năng inject phần phụ thuộc có sẵn sẽ giảm tải công việc lập trình ứng dụng đi rất nhiều. Nó giúp chúng ta có thể nhúng service vào các component hoặc các service với nhau tùy theo nhu cầu của lập trình viên và của ứng dụng.  
- Các chỉ thị ( Directives): Chỉ thị là tính năng mang tính thử thách nhất của Angular. Bạn có thể tạo ra các thẻ HTML tùy biến với vai trò là các widget tùy biến với các chỉ thị. Bạn cũng có thể tùy ý thao tác với các thuộc tính DOM bằng các chỉ thị.

## Giới thiệu mô hình 3 lớp

### Định nghĩa

3-tiers là một kiến trúc kiểu client/server mà trong đó giao diện người dùng (UI-user interface), các quy tắc xử lý (BR-business rule hay BL-business logic), và việc lưu trữ dữ liệu được phát triển như những module độc lập, và hầu hết là được duy trì trên các nền tảng độc lập, và mô hình 3 tầng (3-tiers) được coi là một kiến trúc phần mềm và là một mẫu thiết kế.

### Ưu điểm

1. Việc phân chia thành từng lớp giúp cho code được tường minh hơn. Nhờ vào việc chia ra từng lớp đảm nhận các chức năng khác nhau và riêng biệt như giao diện, xử lý, truy vấn thay vì để tất cả lại một chỗ. Nhằm giảm sự kết dính.
2. Dễ bảo trì khi được phân chia, thì một thành phần của hệ thống sẽ dễ thay đổi. Việc thay đổi này có thể được cô lập trong 1 lớp, hoặc ảnh hưởng đến lớp gần nhất mà không ảnh hưởng đến cả chương trình.
3. Dễ phát triển, tái sử dụng: khi chúng ta muốn thêm một chức năng nào đó thì việc lập trình theo một mô hình sẽ dễ dàng hơn vì chúng ta đã có chuẩn để tuân theo. Và việc sử dụng lại  khi có sự thay đổi giữa hai môi trường ( Winform sang Webfrom ) thì chỉ việc thay đổi lại lớp GUI.
4. Dễ bàn giao. Nếu mọi người đều theo một quy chuẩn đã được định sẵn, thì công việc bàn giao, tương tác với nhau sẽ dễ dàng hơn và tiết kiệm được nhiều thời gian.
5. Dễ phân phối khối lượng công việc. Mỗi một nhóm, một bộ phận sẽ nhận một nhiệm vụ trong mô hình 3 lớp. Việc phân chia rõ ràng như thế sẽ giúp các lập trình viên kiểm soát được khối lượng công việc của mình.

# : PHÂN TÍCH THIẾT KẾ HỆ THỐNG

# : XÂY DỰNG CƠ SỞ DỮ LIỆU

# : XÂY DỰNG GIAO DIỆN VÀ CHẠY THỬ NGHIỆM

# TÀI LIỆU THAM KHẢO

x

|  |  |
| --- | --- |
| [1] | Andrew Tuner, *Introduction to Neogeography*. Sebastopol, CA USA: O'Reilly Media, 2006. |
| [2] | kdtqt.duytan.edu.vn. Tổng quan về mô hình MVC. [Online]. <http://kdtqt.duytan.edu.vn/Home/ArticleDetail/vn/33/1790/tong-quan-ve-mo-hinh-mvc> |
| [3] | microsoft. Tổng quan về ASP.NET MVC. [Online]. <https://support.microsoft.com/vi-vn/help/2645095> |
| [4] | tutorialspoint.com. ASP.NET MVC tổng quan. [Online]. <https://www.tutorialspoint.com/asp.net_mvc/asp.net_mvc_overview.htm> |
| [5] | Lê Văn Nam, *Lập trình.NET 2*. HaNoi: HUMG, 2018. |

x