**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**------------------------------**



**BÁO CÁO**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài:***

**“Xây dựng website tìm phòng trọ cho sinh viên”**

**Người hướng dẫn : THẠC SĨ HUỲNH TRỌNG THƯA**

**Sinh viên thực hiện: ĐẶNG QUANG NHẬT**

**Mã số sinh viên : N12DCCN028**

**Lớp : D12CQPM01**

**Khóa : 2012 – 2017**

**Hệ : ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 11 / 2016**

**LỜI CẢM ƠN**

Để hoàn thành chuyên đề báo cáo tốt nghiệp này trước hết em xin gửi đến quý thầy cô giáo trong khoa Công nghệ thông tin 2 lời cảm ơn chân thành.

Đặc biệt em xin gửi đến giảng viên hướng dẫn, Ths.Huỳnh Trọng Thưa, người đã tận tình hướng dẫn, giúp đỡ em hoàn thành chuyên đề báo cáo tốt nghiệp này lời cảm ơn sâu sắc nhất.

Vì kiến thức bản thân còn hạn chế, việc hoàn thiện đề tài tốt nghiệp này em không tránh khỏi những sai sót, hạn chế, kính mong nhận được những ý kiến đóng góp từ các thầy cô giáo.

Em xin chân thành cám ơn.

Sinh viên thực hiện

**Đặng Quang Nhật**

**MỤc lỤc**

[LỜI MỞ ĐẦU 1](#_Toc468316413)

[CHƯƠNG 1: ĐẶT VẤN ĐỀ 2](#_Toc468316414)

[1.1. Phòng trọ cho sinh viên: 2](#_Toc468316415)

[1.2. Các tiêu chí phòng trọ sinh viên: 2](#_Toc468316416)

[CHƯƠNG 2: MÔ HÌNH HOẠT ĐỘNG WEBSITE 4](#_Toc468316417)

[2.1. Phân tích thiết kế hệ thống: 4](#_Toc468316418)

[2.1.1. Sơ đồ phân rã chức năng: 4](#_Toc468316419)

[2.1.2. Biểu đồ dòng dữ liệu (Data Flow Diagram - DFD) 5](#_Toc468316420)

[2.2. Mô hình hoạt động website: 9](#_Toc468316421)

[CHƯƠNG 3: CÔNG NGHỆ SỬ DỤNG 10](#_Toc468316422)

[3.1. AngularJS: 10](#_Toc468316423)

[3.1.1. AngularJS là gì? 10](#_Toc468316424)

[3.1.2. Kiến trúc AngularJS 2: 11](#_Toc468316425)

[3.2. Công nghệ JAX-RS: 15](#_Toc468316426)

[3.2.1. Cấu trúc REST: 15](#_Toc468316427)

[3.2.2. Dịch vụ web RESTful: 16](#_Toc468316428)

[3.2.3. Cách thức hoạt động của dịch vụ web: 17](#_Toc468316429)

[3.2.4. JAX-RS tạo dịch vụ web RESTful: 17](#_Toc468316430)

[3.3. Hệ quản trị cơ sở dữ liệu MySQL: 17](#_Toc468316431)

[3.3.1. Thế nào là một hệ quản trị cơ sở dữ liệu: 17](#_Toc468316432)

[3.3.2. Đặc điểm của hệ quản trị cơ sở dữ liệu MySQL: 17](#_Toc468316433)

[3.3.3. Stored Procedure trong MySQL: 18](#_Toc468316434)

[3.4. Công cụ quản lý dự án Apache Maven: 19](#_Toc468316435)

[CHƯƠNG 4: TRIỂN KHAI WEBSITE 20](#_Toc468316436)

[4.1. Tổng quan chương trình: 20](#_Toc468316437)

[4.2. Cài đặt chương trình: 20](#_Toc468316438)

[4.2.1. Client: 20](#_Toc468316439)

[4.2.2. Server: 20](#_Toc468316440)

[CHƯƠNG 5: KẾT QUẢ ĐẠT ĐƯỢC 21](#_Toc468316441)

[5.1. Trang chủ: 21](#_Toc468316442)

[5.2. Trang chi tiết phòng trọ 21](#_Toc468316443)

[5.2. Trang tìm kiếm phòng trọ 21](#_Toc468316444)

[5.2. Modal đăng nhập / đăng ký 21](#_Toc468316445)

[5.2. Trang đặt cọc phòng trọ 21](#_Toc468316446)

[5.2. Trang admin 21](#_Toc468316447)

[KẾT LUẬN 21](#_Toc468316448)

[TÀI LIỆU THAM KHẢO 22](#_Toc468316449)

# LỜI MỞ ĐẦU

Bước vào cuộc sống đại học, các tân sinh viên đã phải đối mặt với không ít khó khăn, từ việc sống xa nhà, xa gia đình, làm quen với chỗ ăn, chỗ ở mới,… Trong đó, việc tìm được một chỗ ở tốt, thuận tiện cho việc sinh hoạt hằng ngày và yên tâm cho việc học tập quả thật rất khó khăn.

Từ đó, cùng với sự giúp đỡ của thầy hướng dẫn, em đã tiến hành thực hiện đề tài: “Xây dựng website tìm phòng trọ cho sinh viên” tạo ra một cộng đồng tin cậy cung cấp thông tin về các phòng trọ dành riêng cho sinh viên, đáp ứng tốt nhất cho các tiêu chí của sinh viên. Sinh viên, đặc biệt là các bạn tân sinh viên, và gia đình sẽ không còn những nối lo về việc tìm kiếm chỗ ở, tập trung hoàn toàn vào việc học hành.

Nội dung báo cáo được chia thành 5 chương sau:

Chương 1: Đặt vấn đề

Chương 2: Mô hình hoạt động website

Chương 3: Công nghệ sử dụng

Chương 4: Triền khai website

Chương 5: Kết quả đạt được

# CHƯƠNG 1: ĐẶT VẤN ĐỀ

## Phòng trọ cho sinh viên:

Mỗi năm đầu tháng 9, có thêm hàng triệu tân sinh viên trúng cử vào các trường đại học, thế nhưng phòng trọ, nhà ở thì không “nở” ra thêm được, dẫn tới việc tìm kiếm một phòng trọ phù hợp, ưng ý khó khăn. Không những thế, lợi dụng sự khó khăn này, giới cò mồi và chủ phòng trọ còn đưa ra những chiêu trò nâng giá khá cao, đặc biệt tại các phòng trọ gần trường, thế nhưng cơ sở vật chất thì lại không đáp ứng đủ. Một lựa chọn khác nữa thường được các bạn tân sinh viên chọn là ở ký túc xá của trường. Việc ở ký túc xá trường sẽ bảo đảm cho bạn việc học tập tốt hơn, mọi hoạt động, sinh hoạt của trường bạn đều dễ dàng nắm bắt, thế nhưng vì là một lựa chọn “tốt”, nên rất nhiều sinh viên đăng ký, và mỗi phòng thì bạn phải ở chung với ít nhất là 4 5 người bạn nữa. Về lâu dài có thể bạn sẽ cảm thấy khó khăn vì chung đụng với nhiều người, không có không gian riêng và an ninh thì không được đảm bảo bởi rất khó kiểm soát với số lượng sinh viên quá đông trong từng phòng. Vì thế sinh viên lại có xu hướng tìm phòng trọ, nhưng tìm ở đâu, tìm như thế nào, giá cả ra sao, phải tìm được chỗ thuận tiện cho việc học tập, sinh hoạt, … rất nhiểu vấn đề mà các sinh viên băn khoăn khi tìm một phòng trọ.

## Các tiêu chí phòng trọ sinh viên:

Qua số liệu thu thập được, có một số tiêu chí chung mà các bạn sinh viên thường đề cập tới khi quyết định chọn một phòng trọ :

* Địa điểm: càng gần trường học càng tốt, gần chợ, nơi buôn bán
* Giá cả: hợp lý, tiền điện , nước phù hợp với quy định của nhà nước, sử dụng tự do, dùng bao nhiêu trả bấy nhiêu, có đồng hồ đo điện, nước riêng.
* An ninh trật tự: không có tệ nạn, gần các chốt dân phòng, tránh các hẻm quá tối tăm, vắng người mướn.
* Thuê nguyên phòng hay ở ghép: điều này cũng làm khá nhiều bạn sinh viên băn khoăn bởi giá thuê nguyên phòng thì khá cao, tuy nhiên có không gian riêng tư, còn ở ghép thì giá cả các thứ sẽ được chia ra, tuy nhiên phải sống hòa đồng với nhau. Có một trường hợp thuê nữa là ở chung với chủ, tuy khá ít nhưng hình thức này được nhiều bạn sinh viên quan tâm bởi lý do an ninh.
* Cơ sở vật chất: sạch sẽ, có phòng vệ sinh riêng, có thể thêm gác lửng, đủ không gian để sinh hoạt, học tập.
* Dịch vụ : có nơi để rác, người hốt rác, nơi gửi xe, có wifi, internet, tiền chi trả rõ ràng, minh bạch.
* Bạn bè: việc ở chung với bạn cùng trường, cùng lớp hay cùng niên khóa sẽ dễ dàng hơn cho việc hòa nhập với nơi ở mới, đồng thời giúp ích khá nhiều cho việc học cũng như giờ giấc sinh hoạt.

Qua một số tiêu chí kể trên, ta có thể thấy các điểm khác biệt trong việc tìm kiếm phòng trọ của sinh viên so với những đối tượng khác. Vì thế em muốn tạo ra một website dành riêng cho các bạn sinh viên, với những tiêu chí riêng của mình về phòng trọ, là nơi tin cậy, an toàn cho các bạn trong việc lựa chọn cho mình một nơi ở. Website được xây dựng dựa trên ba công nghệ chủ yếu :

* Angular 2: phiên bản Angular thứ hai với sự thay đổi hoàn toàn và hoàn thiện hơn so với phiên bản đầu tiên, hơn thế dự án Angular được “gã khổng lồ” trong ngành công nghệ Google tài trợ, cộng với một cộng đồng lớn mạnh đã có sẵn hứa hẹn sẽ phát triển nhanh và ưu việt hơn cho website.
* JAX-RS: là một JAVA API hỗ trợ trong việc xây dựng một web service hoàn chỉnh theo ngôn ngữ JAVA, cụ thể ở đây là một RESTful web service.
* MySQL: là hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến nhất hiện nay, giao tiếp dễ dàng với các ứng dụng và trong việc triển khai cơ sở dữ liệu lên server.

Chương tiếp theo đây, em xin trình bày mô hình hoạt động của website.

# CHƯƠNG 2: MÔ HÌNH HOẠT ĐỘNG WEBSITE

## 2.1. Phân tích thiết kế hệ thống:

### 2.1.1. Sơ đồ phân rã chức năng:

Website tìm kiếm

phòng trọ cho sinh viên

**Quản lý người dùng**

**Thống kê**

Đăng ký

Thống kê số tài khoản

Thống kê

giao dịch

Thống kê số phòng trọ

Đăng nhập

**Quản lý phòng trọ**

Xóa

phòng trọ

Cập nhật phòng trọ

Tìm kiếm phòng trọ

Đặt cọc phòng trọ

Tạo

phòng trọ

Duyệt phòng trọ

Ẩn

phòng trọ

Hình 2.1: Sơ đồ phân rã chức năng

### 2.1.2. Biểu đồ dòng dữ liệu (Data Flow Diagram - DFD)

**(\*)** Bộ ký hiệu sử dụng trong quá trình phân tích và thiết kế hệ thống:

**Xử lý**

**Dòng dữ liệu**

**Kho dữ liệu**

**Tác nhân ngoài**

*Hình 2.2 : Các hình vẽ ký hiệu sử dụng trong phân tích thiết kế hệ thống*

#### 2.1.2.1. Mức ngữ cảnh:

Trả kết quả

Quản trị website

Website

Tìm kiếm

Xác nhận đặt cọc

Gửi yêu cầu

Phản hồi

Đặt cọc

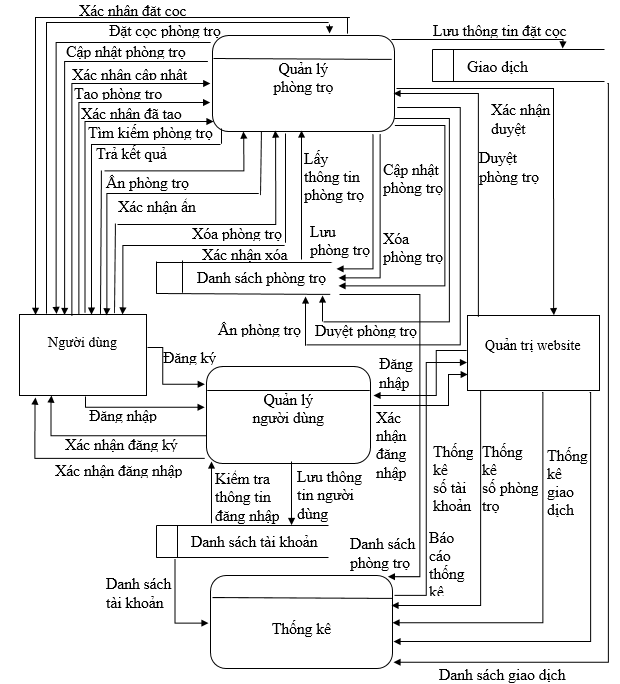
Gửi yêu cầu

Phản hồi

Người dùng

*Hình 2.3: Mức ngữ cảnh*

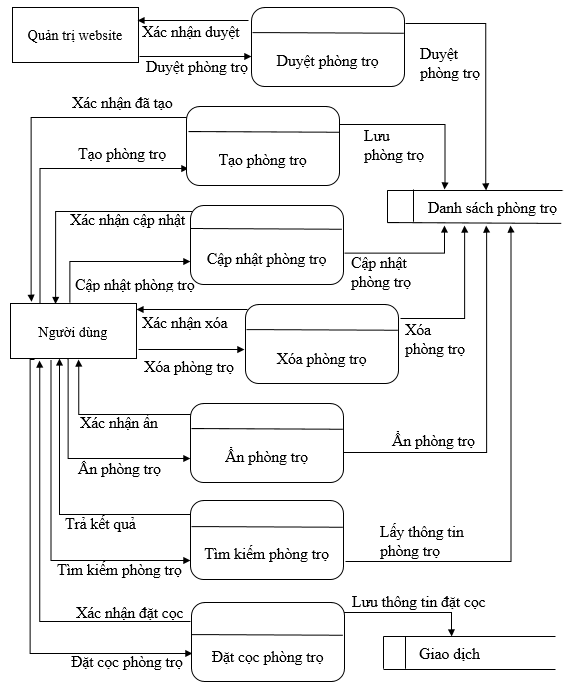
#### 2.1.2.2. Mức 0:



Hinh 2.4: Mức 0

#### 2.1.2.3. Mức 1:

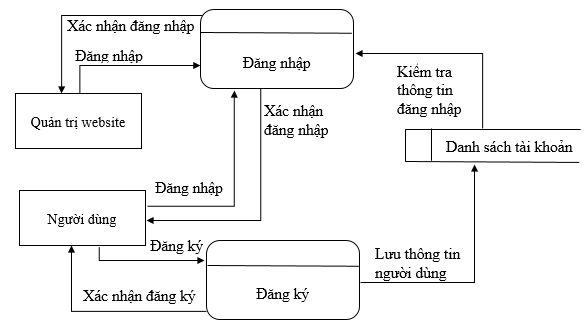
a. Mức 1 – Quản lý phòng trọ



Hình 2.5: Mức 1 – Quản lý phòng trọ

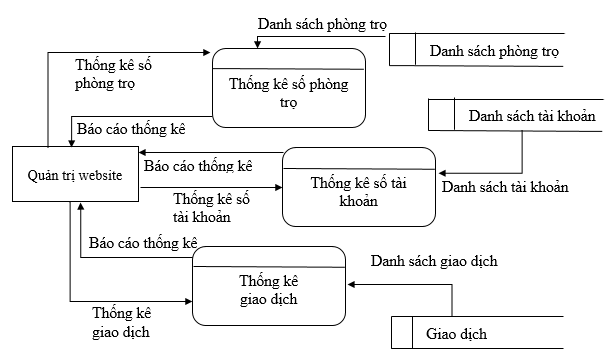
### 

b. Mức 1 – Quản lý người dùng



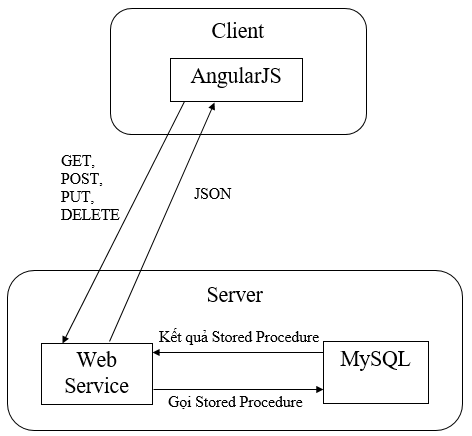
Hình 2.6: Mức 1 – Quản lý người dùng

c. Mức 1 – Thống kê:



Hình 2.7: Mức 1 – Thống kê

## 2.2. Mô hình hoạt động website:



Hình 2.8: Mô hình hoạt động website

Phía client: Framework AngularJS được sử dụng để hiển thị website, đồng thời gởi các gói tin request HTTP với các phương thức GET, POST, PUT, DELETE đến Web service phía server, Web Service sẽ trả dữ liệu về theo dạng JSON.

Phía server: có hai phần là Web Service và hệ quản trị cơ sở dữ liệu MySQL. Trong đó Web Service sẽ tiến hành gọi các Stored Procedure trong MySQL để lấy dữ liệu cần thiết, đóng gói dữ liệu theo kiểu JSON và gửi đến cho client.

Chương tiếp theo đây sẽ nói rõ hơn về các công nghệ được sử dụng trong quá trình xây dựng website tìm kiếm phòng trọ cho sinh viên.

# CHƯƠNG 3: CÔNG NGHỆ SỬ DỤNG

## 3.1. AngularJS:

### 3.1.1. AngularJS là gì?

AngularJS là một framework được bảo trợ bởi Google cho phép xây dựng các ứng dụng phía client bằng HTML và Javascript. Trong khi HTML được thiết kế nhằm xây dựng các chương trình tĩnh, website tĩnh, thì AngularJS chính là công cụ cần thiết mà HTML hướng tới để xây dựng các chương trình , website động.

Vậy thế nào là một website tĩnh, và thế nào là một website động ?

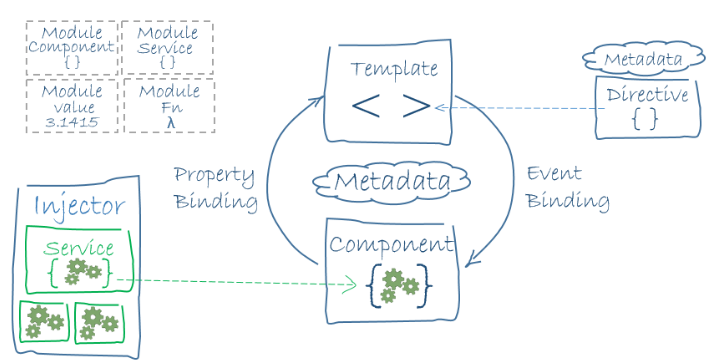
Một website tĩnh là một trang web mà nội dung truyền tải của nó tới người dùng sẽ không thay đổi một khi đã được tạo.

Một website động là một trang web mà nội dung của nó sẽ được thay đổi mỗi khi người dùng truy cập tới.

Để tạo một website động từ website tĩnh được xây dựng từ HTML, Javascript được thiết kế để làm điều đó, cùng với đó là việc sử dụng kết hợp nhiều thư viện hay framework khác nhau như : Jquery, Ember. Thế nhưng với AngularJS, nó thay đổi và thêm mới trực tiếp vào cấu trúc của HTML, AngularJS tạo ra các “directive” dùng để chỉ thị việc hiển thị nội dung động một cách dễ dàng, AngularJS tạo ra các “service” thực hiện các logic của chương trình, đồng thời giao tiếp với server, AngularJS tạo ra các “component” để nhóm lại các thành phần liên quan với nhau giúp cho việc lập trình rõ ràng, dễ sử dụng, dễ kiểm tra, dễ triển khai, dễ bảo trì và dễ dàng cho việc sử dụng lại.

Hiện nay AngularJS được phát triển lên phiên bản 2, tuy nhiên phiên bản 2 này lại không hoàn toàn là bản nâng cấp từ phiên bản 1, mà gần như thay đổi hoàn toàn cấu trúc cũng như nâng cao đáng kể về hiệu suất. AngularJS 2 được phát triển với mục tiêu chạy trên mọi nền tảng, đặc biệt hướng tới mobile trong khi AngularJS 1 thì không, AngularJS 2 được xây dựng với nhiều ngôn ngữ hơn không chỉ Javascript thuần nữa mà mở rộng hơn với TypeScript, Dart , hoặc Javascript viết theo chuẩn ES5, ES6, nhờ đó AngularJS 2 được hỗ trợ tốt hơn từ các IDE và plugin được xây dựng cho các ngôn ngữ này. Trong đề tài này, em xin trình bày phiên bản 2 của AngularJS mà bỏ qua phiên bản 1 do sự khác biệt rất lớn này.

### 3.1.2. Kiến trúc AngularJS 2:



Hình 3.1: Kiến trúc AngularJS 2

Kiến trúc AngularJS 2 được xây dựng dựa trên 8 thành phần chính:

- Modules

- Components

- Templates

- Metadata

- Data binding

- Directives

- Services

- Dependency injection

Modules: Angular Module là một class thông thường được gắn với decorator @NgModule.

Decorator @NgModule được sử dụng nhằm khai báo cho AngularJS biết rằng class này được xử lý như thế nào so với các phần tử khác cũng đều là một class thông thường trong Javascript như : directive, pipe, component,… Decorator là một phương thức dùng metadata để thực thi. Khi một class được gán decorator @NgModule, thì phải khai báo thêm các thuộc tính cần thiết trong metadata mà @NgModule dùng để phân biệt các Module với nhau, các thuộc tính gồm:

- declarations: gồm các component, directive và pipe thuộc module.

- imports: gồm các module khác bên ngoài cần sử dụng thêm.

- exports: tập con của declarations, dùng để khai báo các thành phần được public để các module khác có thể sử dụng được.

- providers: gồm các service được các component sử dụng trong modules.

- bootstrap: xác định component nào sẽ là component bắt đầu.

Module được dùng để chứa tất cả thành phần của một ứng dụng gọi là root module, và tất cả các ứng dụng đều có ít nhất một module này, trong đó có thể bao gồm nhiều module nhỏ hơn bao gồm các component, directive , service có liên quan đến nhau, đến một logic, đến một trang của website,… nhằm giúp cho việc tái sử dụng dễ dàng, các đoạn code rõ ràng hơn, dễ thay đổi, dễ kiểm tra, dễ đọc.

Components: quản lý một “view” của webiste, vẫn là một class thông thường trong Javascript và được gán với decorator @Component, mỗi decorator @Component khai báo cần một số thuộc tính sau:

- Selector: tag tùy chỉnh AngularJS, là nơi đặt template của component vào đó.

- Template: được viết dựa trên HTML, AngularJS dựa trên template để hiển thị view.

- Style: chứa các CSS để trang trí cho view

Template và các biến, phương thức được khai báo trong Class Component sẽ tương tác với nhau thông qua Data Binding.

Metadata: một thành phần quan trọng, nó cho AngularJS biết cách thức xử lý với các class thông thường với vai trò khác nhau, ví dụ như Module, Component, Service, …. Tất cả đều là class Javascript thông thường cho đến khi được gán thêm với decorator lên chúng, và decorator sẽ dùng các metadata để định nghĩa về class đó. Ví dụ về metadata:

@Component({

selector: ‘app-test’,

templateUrl: ‘app-test.component.html’,

styleUrls: [‘app-test.component.css’]

})

Trong đó:

- selector: ghi theo selector của CSS báo cho AngularJS vị trí sẽ đặt Component, cụ thể là template vào đó, nghĩa là với mỗi nơi có cặp tag <app-test></app-test> thì Component sẽ được đặt vào trong nó.

- templateUrl: địa chỉ tương đối dẫn đến file HTML template của Component.

- styleUrls: các địa chỉ tương đối dẫn tới các file CSS của template.

Data binding: như đã nói ở trên, Template sẽ giao tiếp với các biến, các phương thức trong Component thông qua dịch vụ Data Binding. Data binding gồm hai dạng chính là một chiều và hai chiều, trong đó một chiều có các dạng sau:

Template -> Component: truyền dữ liệu từ Template vào một biến của component hoặc gọi một phương thức trong Component, ví dụ : (click)=”handle()”, được đặt như các thuộc tính của một tag HTML,

Component -> Template: truyền dữ liệu từ Component tới Template, có hai cách để Component có thể truyền dữ liệu cho Template:

- Sử dụng hai dấu ngoặc nhọn: {{var}}, trong đó var là một biến được khai báo trong Component.

- Property Binding: với những thuộc tính của một tag HTML thông thường như value, id, src,… Ta chỉ cần thêm dấu dấu ngoặc vuông [value]=”var”, với biến var được khai báo trong Component thì thuộc tính value của tag HTML sẽ có giá trị của biến var.

Data binding hai chiều có dạng như sau:

Template <--> Component: đặt [(ngModel)]=”var” với biến var được khai báo trong Component trong một số tag HTMl như input, textarea để có thể lấy được giá trị của tag đó trong Component theo thời gian thực mỗi khi người dùng thay đổi giá trị của phần tử đó.

Directives: là một class được gán với decorator @Directive, Component thật ra chính là một Directive có thêm template. Ngoài Component, Directive còn có hai dạng nữa là : structural và attribute directives.

- Structural directive: thay đổi layout bằng cách thêm, xóa hoặc sửa các phần tử trong cây DOM của HTML.

Ví dụ: <li \*ngIf=”selectedLi”></li>

\*ngIf là một structural directive được xây dựng sẵn của AngularJS, mục đích là để thêm hoặc xóa phần tử trong DOM.

- Attribute directive:

Ví dụ: <img [src]=”srcImage”>

Attribute directive trông giống attribute của các tag HTML với thêm hai dấu ngoặc vuông, directive này thường dùng để lấy giá trị từ các biến trong Component gán vào giá trị các thuộc tính của tag HTML, nhờ đó các giá trị của thuộc tính có thể được thay đổi thường xuyên.

Services: bất kỳ một phương thức, tính năng nào mà chương trình cần đều có thể là service. Service là một class thông thường, được tạo ra nhằm phục vụ một mục đích nhất định và duy nhất. Ví dụ: logging service, data service.… Service thường được gán với decorator @Injector với mục đích sử dụng cho Dependency Injection.

Dependency Injection (DI): DI giúp tạo instance của một class mà không cần phải khai báo new. Đa số được sử dụng với các Service. Bằng cách gán decorator @Injectable lên class Service, mỗi khi Component cần sử dụng Service này, ta chỉ cần khai báo ở tham biến của hàm constructor của Component là đã có thể sử dụng được, ví dụ: constructor(private service: Service) - khai báo biến service với kiểu là class Service, khi đó ở bất kỳ nơi nào trong Component, ta đều có thể sử dụng ngay service này mà không cần phải tạo instance cho nó nữa (nghĩa là service = new Service()).

Ngoài 8 thành phần chính, trong đề tài này, em còn sử dụng thêm một số tính năng khác như: pipe, router, http service.

Pipe: dùng để định dạng lại giá trị hiển thị trên template HTML, Ví dụ: {{varDate | date:”dd/MM/yy”}} - | được gọi là pipe operator, và date là một pipe được định nghĩa sẵn từ AngularJS, với biến varDate là một biến trong Component, giả sử biến varDate có giá trị là April 10, 1998, khi sử dụng pipe với định dạng hiển thị là dd/MM/yy, thì giá trị của biến varDate sẽ được hiển thị : 10/04/98.

Router: dùng trong việc điều hướng giữa các view trong website với nhau, router định nghĩa từng đường dẫn tới mỗi view, tương ứng với một Component xác định, ví dụ :

const appRoutes: Routes = [{

path: ‘test’, component: testComponent

}];

Http Service: dùng để lấy dữ liệu từ server, cụ thể ở đây là sử dụng các phương thức GET, POST, PUT, DELETE có trong API của Web Service để lấy dữ liệu hiển thị lên website.

Tổng hợp lại, với kiến trúc đã được trình bày như trên, phía client của website được tạo như sau:

- Để lấy dữ liệu, giao tiếp với server: Http Service và một số service tự tạo thực hiện logic trả về dữ liệu.

- Để hiển thị dữ liệu lên website:

+ Template: được viết bằng HTML và thêm các cú pháp mới được AngularJS biên soạn thêm để giao tiếp trực tiếp với Component, nhằm lấy giá trị cũng như cập nhật giá trị các biến trong Component, nhờ vào cơ chế Data binding.

+ Style: là các CSS được viết để trang trí cho từng template.

- Để đóng gói từng view, từng trang của website, ta sử dụng Component. Mỗi Component chứa một template, nhiều style cho template đó, và một selector nơi đặt template.

- Các Component sẽ sử dụng các Service, và nhờ cơ chế DI giúp cho việc sử dụng dễ dàng hơn, không còn phải khai báo các instance của Service mỗi khi dùng, chỉ việc khai báo một lần trong dãy tham biến của hàm constructor.

- Để điều hướng qua lại giữa các Component, các view, ta sử dụng Angular Router.

- Modules được dùng để đóng gói tất cả các Component của một chương trình, được dùng để khai báo các Service được sử dụng chung cho toàn bộ Component, được dùng để khai báo Component cha được khởi tạo đầu tiên.

- Và cuối cùng là Metadata đi theo với các Decorator nhằm xác định các class được tạo ra , class nào là Module, class nào là Component, class nào là Directive,…

## 3.2. Công nghệ JAX-RS:

Công nghệ JAX-RS được xây dựng nhằm tạo ra các RESTful Web Service, được viết theo ngôn ngữ Java. Để hiểu được cấu trúc, cách vận hành của RESTful Web Service, trước hết ta đi tìm hiểu thế nào được gọi là một hệ thống REST.

### 3.2.1. Cấu trúc REST:

Cấu trúc REST (Representational State Transfer): sử dụng phương thức HTTP để giao tiếp, truyền thông tin. Cấu trúc này coi mọi thứ đều là tài nguyên, và các tài nguyên sẽ được truy xuất thông qua các giao thức chuẩn của HTTP (GET, POST, PUT, DELETE,…).

Khi áp dụng đầy đủ các ràng buộc và quy ước sau vào hệ thống, ta sẽ có một hệ thống REST, hay gọi là hệ thống RESTful:

* + Hoạt động theo mô hình client – server: client và server được phát triển độc lập với nhau, giao tiếp với nhau thông qua các interface định sẵn, mỗi tài nguyên sẽ được chỉ định bởi một URI do server cung cấp.
  + Phi trạng thái: server sẽ không lưu bất kỳ thông tin, trạng thái nào của client, vì thế mỗi khi client có yêu cầu sử dũng một dịch vụ của server thì phải đóng gói đầy đủ các thông tin cần thiết của mình về cho server.
  + Có khả năng cache: hệ thống có thiết bị trung gian đóng vai trò lưu trữ tạm các dữ liệu đã được yêu cầu trước đó, vì thế client có thể lấy dữ liệu một cách nhanh chóng, đồng thời giảm tần suất gọi dịch vụ trên server của client.
  + Tính phân lớp: Hệ thống được chia ra thành từng lớp, trong đó các lớp kề nhau thì giao tiếp với nhau, không giao tiếp với các lớp khác, các lớp dưới thường cung cấp dịch vụ phục vụ cho các lớp trên.
  + Có các giao tiếp thống nhất: đây là ràng buộc quan trọng và cơ bản của một hệ thống REST, sự thống nhất về các giao tiếp giúp client và server có thể được phát triển độc lập với nhau, dù cho client là mobile, web, hay một thiết bị nhúng bất kỳ đều có thể gọi dịch vụ trên server thông qua các giao tiếp thống nhất này.

Một hệ thống RESTful phải có các ràng buộc ở trên, tuy nhiên đối với một dịch vụ web RESTful ngoài những ràng buộc ở trên, dịch vụ web còn phải đáp ứng một số nguyên tắc thiết kế khác sẽ được trình bày trong phần tiếp theo đây.

### 3.2.2. Dịch vụ web RESTful:

Một dịch vụ web RESTful cần tuân thủ theo bốn nguyên tắc thiết kế sau đây:

* + Sử dụng các phương thức chuẩn của HTTP một cách rõ ràng: các phương thức chuẩn của HTTP như GET, POST, PUT, DELETE,… phải được sử dụng theo đúng định nghĩa bởi RFC 2616. Nguyên tắc thiết kế này thiết lập một ánh xạ 1-1 giữa các phương thức của HTTP với các thao tác tạo, đọc, sửa và xóa dữ liệu (CRUD):
    - GET: truy xuất một tài nguyên.
    - POST: tạo mới một tài nguyên.
    - PUT: cập nhật, thay đổi trạng thái của một tài nguyên.
    - DELETE: xóa một tài nguyên.
  + Hệ thống phi trạng thái: nhằm đáp ứng ràng buộc của một hệ thống REST.
  + Hiển thị cấu trúc thư mục giống với các URI: Cấu trúc các thư mục, nơi đặt các tài nguyên, phải được sắp xếp theo đúng đường dẫn URI mà server chỉ định cho tài nguyên đó. Một số lưu ý về cấu trúc địa chỉ URI:
    - Chữ thường.
    - Thay thế khoảng trống bằng gạch nối (-) hay gạch chân ( \_ ).

Sự chuyển đổi sang JSON (Javascript Object Notation) và XML: các dịch vụ sẽ trả về kết quả cho client kiểu dữ liệu thuộc dạng JSON hoặc XML, mặc dù dữ liệu khi được lấy lên từ database không thuộc cả JSON và XML. Điều này giúp cho các ứng dụng, thiết bị bên client, được lập trình với mọi ngôn ngữ, đều có thể sử dụng dịch vụ và lấy được dữ liệu.

### 3.2.3. Cách thức hoạt động của dịch vụ web:

Dịch vụ web là nơi trung gian luân chuyển dữ liệu giữa hệ quản trị cơ sở dữ liệu MySQL và phía client, cụ thể là AngularJS.

Đối với phía client, dịch vụ web cung cấp các API là các phương thức GET, POST, PUT, DELETE của HTTP cho AngularJS thực hiện các yêu cầu, sau đó gửi trả về dữ liệu dưới dạng JSON.

Đối với MySQL, dịch vụ web nhận yêu cầu từ phía client, thực hiện lệnh gọi các Stored Procedure tương ứng trong MySQL, MySQL trả về dữ liệu từ đó đóng gói dưới dạng JSON trả về cho client.

### 3.2.4. JAX-RS tạo dịch vụ web RESTful:

JAX-RS cung cấp các annotation cho phép khai báo một cách nhanh chóng các phương thức của một class tương ứng với các phương thức HTTP như GET (annotation là @GET), POST (@POST),… Ngoài ra còn một số annotation riêng như @Path định nghĩa đường dẫn tương đối đến phương thức hoặc cả một class, @Produces định nghĩa dạng dữ liệu sẽ được trả về cho phía client, thường có hai dạng là JSON và XML.

## 3.3. Hệ quản trị cơ sở dữ liệu MySQL:

### 3.3.1. Thế nào là một hệ quản trị cơ sở dữ liệu:

Hệ quản trị cơ sở dữ liệu là một phần mềm được dùng để thiết kế và quản trị cơ sở dữ liệu, hỗ trợ cho người dùng trong việc tạo, lấy, thay đổi và quản lý dữ liệu.

### 3.3.2. Đặc điểm của hệ quản trị cơ sở dữ liệu MySQL:

MySQL là một hệ quản trị cơ sở dữ liệu mã nguồn mở, được phát triển và hỗ trợ bởi Oracle.

MySQL sử dụng cơ sở dữ liệu quan hệ, sử dụng ngôn ngữ SQL làm ngôn ngữ truy xuất dữ liệu.

MySQL được viết bằng ngôn ngữ C và C++.

MySQL hỗ trợ nhiều hệ điều hành khác nhau: Linux, Windows, Mac OS,…

MySQL hỗ trợ nhiều ngôn ngữ lập trình : Java, C, C++, PHP, Perl, Python và Tcl.

MySQL hỗ trợ chuẩn giao tiếp ODBC dành cho các ngôn ngữ như Delphi, Visual Basic, ngoài ra tạo riêng chuẩn giao tiếp cho .NET là Connector/NET

MySQL hỗ trợ Trigger, View, Cursors.

MySQL hỗ trợ SubSelect.

MySQL hỗ trợ Full-text search.

MySQL hỗ trợ Replication.

MySQL hỗ trợ Query caching.

MySQL hỗ trợ Stored Procedure với cách viết tuân thủ theo SQL/PSM (tiêu chuẩn ISO về Stored Procedure)

MySQL hỗ trợ quá trình roll back , commit và crash recovery khi sử dụng transactions.

### 3.3.3. Stored Procedure trong MySQL:

Stored Procedure là một nhóm các câu lệnh Transact-SQL đã được biên dịch và chứa trong MySQL dưới một tên nào đó, được xử lý như một đơn vị (chứ không còn là thực thi nhiều câu lệnh riêng lẻ).

Cú pháp tạo một Stored Procedure trong MySQL

CREATE PROCEDURE `mysp\_test`(IN \_varInput int, OUT \_varOutput varchar(45), INOUT \_varIO int)

BEGIN

SELECT column3 into \_varOutput

FROM table

WHERE column1 = \_varInput or column2 = \_varIO;

END ;;

DELIMITER ;

Trong đó :

- “mysp\_test” là tên Stored Procedure.

- Các tham số \_varInput, \_varOutput và \_varIO kiểu dữ liệu lần lượt là int, varchar(45) và int, với tham số được quy định là IN, hay INOUT Stored Procedure có thể lấy giá trị các tham số này sử dụng vào các câu truy vấn SQL, tuy nhiên nếu muốn truyền giá trị ra ngoài thì chỉ có thể select hoặc set giá trị cho các biến OUT hoặc INOUT.

- Các câu lệnh truy vấn sẽ được viết toàn bộ trong khối BEGIN và END.

Để thực thi các Stored Procedure trong MySQL, ta dùng lệnh call mysp\_test(@input, @output, @io), khi thực hiện xong, ta có thể SELECT @output và @io để lấy dữ liệu thực hiện tiếp vì @output là OUT tham số và @io là INOUT tham số.

## 3.4. Công cụ quản lý dự án Apache Maven:

Apache Maven là công cụ giúp quản lý dự án bằng cách đơn giản hóa công việc build dự án, sử dụng khái niệm Project Object Model (POM), từ đó xây dựng cách thức build, các dependencies, modules cần thiết cho dự án.

Maven xây dựng một repositories riêng lưu trữ nhiều thư viện, dependencies giúp lập trình viên không cần phải tìm kiếm ở khắp nơi, chỉ cần khai báo tất cả trong một file POM, trong quá trình build dự án Maven sẽ tự động tải và thiết lập các cài đặt cần thiết để dự án chạy đúng theo mong muốn.

# CHƯƠNG 4: TRIỂN KHAI WEBSITE

## 4.1. Tổng quan chương trình:

|  |  |
| --- | --- |
|  | Nội dung |
| Ứng dụng | Web Application |
| Ngôn ngữ lập trình | HTML, CSS, Javascript, Java |
| Server | Apache Tomcat 8 |
| IDE | Eclipse Mars.2 Release (4.5.2) |
| Cơ sở dữ liệu | MySQL Server 5.7, MySQL Workbench 6.3 CE |
| Công cụ | NPM, Angular CLI, Maven 2 |
| Framework, Thư viện | AngularJS 2, Jquery 3, Bootstrap 3, auth0-lock 10.5.0, chartJS 2.4.0, flexslider 2.6.3, font-awesome 4.6.3, ng2-bootstrap 1.1.14, ng2-charts 1.4.1, ng2-nouislider 0.8/0, nouislider 8.5.1, ng2-uploader 1.2.0, rxjs 5.0.0, JAX-RS |
| Chức năng | 1. Thêm/Xóa/Sửa phòng trọ 2. Duyệt / Không chấp nhận phòng trọ 3. Ẩn / hiện phòng trọ. 4. Đặt cọc phòng trọ. 5. Tra cứu phòng trọ theo tiêu chí của sinh viên. 6. Đăng ký/ Đăng nhập, phân quyền người dùng. 7. Thống kê số liệu về phòng trọ, tài khoản và giao dịch. |

## 4.2. Cài đặt chương trình:

### Client:

### Server:

Có thể thấy việc trồng cây trong nhà kính mang lại rất nhiều lợi ích như: sự tiện lợi, tự động hóa cao, bảo vệ cây trồng mùa màng khỏi sâu bệnh, tai ương, thiên tai tốt, có thể trồng các loại rau củ quả trái mùa mà năng suất vẫn cao, giảm chiCó thể thấy việc trồng cây trong nhà kính mang lại rất nhiều lợi ích như: sự tiện lợi, tự động hóa cao, bảo vệ cây trồng mùa màng khỏi sâu bệnh, tai ương, thiên tai tốt, có thể trồng các loại rau củ quả trái mùa mà năng suất vẫn cao, giảm chiCó thể thấy việc trồng cây trong

# CHƯƠNG 5: KẾT QUẢ ĐẠT ĐƯỢC

## 5.1. Trang chủ:

## 5.2. Trang chi tiết phòng trọ

## 5.2. Trang tìm kiếm phòng trọ

## 5.2. Modal đăng nhập / đăng ký

## 5.2. Trang đặt cọc phòng trọ

## 5.2. Trang admin

# KẾT LUẬN

Có thể thấy việc trồng cây trong nhà kính mang lại rất nhiều lợi ích như: sự tiện lợi, tự động hóa cao, bảo vệ cây trồng mùa màng khỏi sâu bệnh, tai ương, thiên tai tốt, có thể trồng các loại rau củ quả trái mùa mà năng suất vẫn cao, giảm chi phí lao công. Việc trồng cây trong nhà kính ở nước ta hiện nay đang dần phổ biến và được ưa chuộng nhiều hơn. Vì thế việc đầu tư và phát triển hệ thống thông tin môi trường của cây trồng trong nhà kính được quan tâm và chú trọng khá nhiều, bởi nó ảnh hưởng tới sự tự động hóa, một trong những đặc điểm được đánh giá cao của việc trồng cây trong nhà kính với trồng cây ngoài vuờn.

Với một hệ thống thông tin môi trường của cây trồng trong nhà kính, ta có khá nhiều hướng mở rộng, điển hình như xây dựng một ứng dụng trên mobile theo dõi và ghi nhận các họa động của nhà kính, môi trường hiện tại bên trong, thông tin chi tiết về cây trồng, ngày trồng, cây phát triển như thế nào, các thống kê so sánh với các vụ mùa trước đây, người trồng có thể điều khiển các thiết bị trong nhà kính theo ý muốn, ở mọi nơi, mọi lúc miễn là có mạng, ngoài ra ứng dụng còn có thể giúp cho người mua biết các thông tin về nơi sản xuất như được trồng ngày nào, trồng ra sao, xài các loại phân bón gì, thu hoạch ngày nào,… Không những xây dựng một ứng dụng trên mobile, ta có thể xây dựng thêm trang web giúp người trồng theo dõi tốt hơn khi đang ở nhà hoặc sử dụng mobile nhưng không cài ứng dụng.

# TÀI LIỆU THAM KHẢO

1. IBM Bluemix Docs: <https://console.ng.bluemix.net/docs/>
2. IBM IOT Foundation documentation: <https://docs.internetofthings.ibmcloud.com/>
3. NodeMCU documentation: <https://nodemcu.readthedocs.io/en/master/>
4. IBM developerWorks: http://www.ibm.com/developerworks/
5. Series MQTT Essentials: http://www.hivemq.com/mqtt-essentials/