**ANGULAR**

**Tổng quan về angular:**

1. **Angular là gì?**

* Là một javascript framework dùng để xây dựng ứng dụng tại client-side Dùng HTML,CSS và Javascript.

1. **Tại sao dùng angular?**

* HTML tách biệt.
* Gắn kết dữ liệu tốt.
* Thiết kế theo module.
* Làm việc tốt với back-end.

1. **Cài đặt môi trường angular:**

* Cài đặt angular CLI: npm install -g @angular/cli
* Tạo project: ng new projectName
* Thực thi ứng dụng: ng serve -o

**Component:**

1. **Component là gì?**

* Component là một lớp nó thường hiểu như là controller tương tác giao diện người dùng. Một component thường có 3 thành phần chính là code typescript, template HTML và CSS.

1. **Cách tạo 1 component:**

* Để tạo 1 component bạn dùng câu lệnh sau:

**ng generate component[-t][-s] TênComponent**

**Các option của lệnh:**

* -t (-inline-template): Không tạo file html, dùng template ngay trong component
* -s (-inline-style): Không tạo file css, sử dụng css trực tiếp trong component

**Data binding:**

* Dùng để truyền dữ liệu qua các component và từ component ra view.

**Module:**

* Angular cung cấp sẵn nhiều module, các module này được gọi là NgModules.
* Ví dụ: FormsModule, RouterModule, HttpClientModule,…
* Mỗi ứng dụng có 1 module đặc biệt là AppModule, đây là module gốc được dùng để khởi chạy ứng dụng.

**Service:**

* Service là các đoạn code được tạo ra để phục vụ một chức năng nào đó. Chẳng hạn như tính toán, lấy dữ liệu từ server, validate dữ liệu,…. Service được sử dụng trong component thông qua dependency injection. Mỗi service lưu trong file tên Service.service.ts.

**Vòng đời của component trong angular:**

1. **ngOnChanges:**

* Trong component của bạn, nếu có định nghĩa hàm **ngOnChanges**thì hàm này sẽ chạy khi component dò thấy giá trị được binding vào component bằng phương pháp @Input có thay đổi.
* Hàm **ngOnChanges** có 1 tham số kiểu SimpleChange, thông qua tham số này, bạn có thể truy xuất được giá trị mới, giá trị cũ trước và sau thay đổi.

1. **ngOnInit:**

* Trong component, nếu bạn có định nghĩa hàm **NgOnInit**thì hàm này sẽ được gọi duy nhất 1 lần khi component được tạo, nó chạy sau khi hàm constructor và hàm ngOnchange (lần đầu). Trong hàm này bạn có thể thực hiện các công việc cần dùng khi conponent khởi tạo, như gán giá trị khởi đầu cho  các biến, gọi api, ghi log, khởi tạo form…

1. **ngDoCheck:**

* Trong component, nếu bạn có định nghĩa hàm **ngDoCheck** thì hàm này sẽ chạy mỗi khi component phát hiện ra có sự thay đổi dữ liệu ở component. Hàm ngDoCheck() thực thi ngay sau hàm [ngOnChanges()](https://www.stackchief.com/blog/ngOnChanges%20Example%20%7C%20Angular) và [ngOnInit()](https://www.stackchief.com/blog/ngOnInit%20Example%20%7C%20Angular).

1. **ngAfterContentInit:**

* Nếu trong component bạn có định nghĩa hàm **ngAfterContentInit** thì nó sẽ chạy (1 lần) sau khi nội dung component đã được xây dựng thành công trong DOM , tức nội dung component đã được nạp vào trang web lần đầu thành công.

1. **ngAfterContentChecked:**

* Trong component của bạn nếu có hàm **ngAfterContentChecked** thì nó sẽ chạy mỗi lần Angular dò thấy có sự thay đổi trong targets DOM content.
* Hàm này chạy sau hàm **ngAfterContentInit** , nó có thể chạy nhiều lần tùy theo sự thay đổi dữ liệu trong DOM và có thể làm chậm hệ thống nếu bạn lạm dụng quá.

1. **ngAfterViewInit:**

* Hàm này tương tự như **ngAfterContentInit** nhưng nó chạy khi component và các component con của nó được khởi tạo thành công. Chỉ được gọi 1 lần sau khi ngAfterContentChecked.

1. **ngAfterViewChecked:**

* Hàm **ngAfterViewChecked**được gọi chạy sau khi Angular dò thấy có sự thay đổi trong view của component cha và các view của component con.

1. **ngOnDestroy:**

* Trong component, nếu có hàm **ngOnDestroy**thì nó sẽ được gọi chạy 1 lần duy nhất trước khi component bị huỷ bởi Angular. Dùng hàm này nếu cần để hủy connection, unsubribe, xóa các biến trong stored…

**Data Binding Trong Angular:**

1. **Bind dữ liệu từ component ra view:**

* Là cách gắn 1 biến hay biểu thức ra 1 vị trí nào đó trong view. Nói đúng hơn là gắn vào trong 1 element nào đó của DOM. Angular sẽ đưa ra element đó giá trị của biến/biểu thức khi có thay đổi .
* Cú pháp như sau: {{tên\_biến/biểu thức}}

1. **Bind dữ liệu từ component ra thuộc tính của tag HTML:**

* Cú pháp bind thuộc tính như sau:

<tag [thuoctinhHTML]=”tênbiến”>

1. **Bind sự kiện trong DOM với hàm trong component:**

* Cú pháp để gắn là:

<tag (tênsựkiện)=“tên hàm()”>

**Định dạng dữ liệu với Angular Pipe:**

1. **Định dạng biến kiểu string với angular pipe:**

* Các biến kiểu string khi hiển thị trong view có thể điều chỉnh lại. Gồm dạng chữ hoa, chữ thường, hoặc chữ hoa đầu mỗi từ. Cách dùng như sau:

{{tenbien|uppercase}} hoặc {{tenbien|lowercase}} hoặc {{tenbien|titlecase}}

1. **Định dạng biến kiểu ngày giờ với angular pipe:**

* Các biến có kiểu ngày giờ khi hiển thị trong view có thể định dạng lại theo format mong muốn . Cú pháp là:

{{TênBiến | date:format}}

1. **Định dạng số với angular pipe:**

* Cú pháp lệnh để định dạng số là như sau:

**{{ tênbiến|number[:digitsInfo[:locale]]}}**

1. **Định dạng tiền tệ với angular pipe:**

* Cú pháp định dạng tiền tệ :

**{{tênbiến|**[**currency**](https://angular.io/api/common/CurrencyPipe)**[:currencyCode[:display[:digitsInfo[:locale]]]]}}**

**Các chỉ thị thường dùng trong angular:**

1. **Chỉ thị \*ngIf trong angular:**

* Sử dụng chị thị \*ngIf bằng cách ghi **\*ngIf=”Điềukiện”** trong 1 tag của view sẽ giúp tạo/không tạo tag đó trong DOM tùy theo điều kiện đúng hay sai.

1. **Chỉ thị \*ngFor trong angular:**

* Sử dụng chỉ thị **\*ngFor** trong 1 tag của view sẽ giúp tạo ra tag đó nhiều lần trong DOM.

1. **Chỉ thị ngSwitch:**

* Đây là chỉ thị rẽ nhánh, tương đương nhiều lệnh \*ngIf

**Routing trong angular:**

* Routing trong Angular là chức năng giúp nạp component vào trang web theo từng route (đường dẫn) đã khai báo, đây là chức năng nền tảng để tạo nên Single Page Application – ứng dụng web 1 trang.

**Làm việc với form trong angular:**

* Có hai cách để làm việc với form trong Angular, đó là template-driven và reactive. Hai cách này thực hiện khác nhau khi xử lý dữ liệu và sự kiện trong form. Mỗi cách có những thuận lợi riêng.

**Sử dụng http service trong angular:**

* Sử dụng http service trong angular để tương tác với server nhằm mục đích request lấy tài nguyên hay các thao tác phía server. Service http rất dễ dùng. Với nó, bạn có thể tạo request dạng get, post, put, delete để phục vụ cho nhiều mục đích khác nhau.

1. **Inport http service vào project:**

* Để dùng được service, bạn import nó vào project của mình. Bằng cách khai báo trong file app.modules.ts

**Authentication trong angular:**

* Authentication trong Angular là xác thực người dùng, nhằm xác định người đang dùng ứng dụng là ai để cấp quyền truy cập.
* Liên quan đến authentication, có một số việc cần làm cơ bản sau đây: Xử lý đăng nhập của user, chức năng thoát, nhận thông tin profile của người dùng, bảo vệ ứng dụng qua các đường route và gọi hàm tại các vị trí cần thiết để kiểm tra.
* Để thực hiện Authentication trong Angular, bạn cần biết các loại route guard  và cách tạo. Route guard là bảo vệ các route trong ứng dụng. User được phép vào 1 route nào đó hay không tùy thuộc vào sự đánh giá true/false của guard. Trong Angular có các loại route guard là: CanActivate,  CanActivateChild, CanDeactivate, CanLoad.