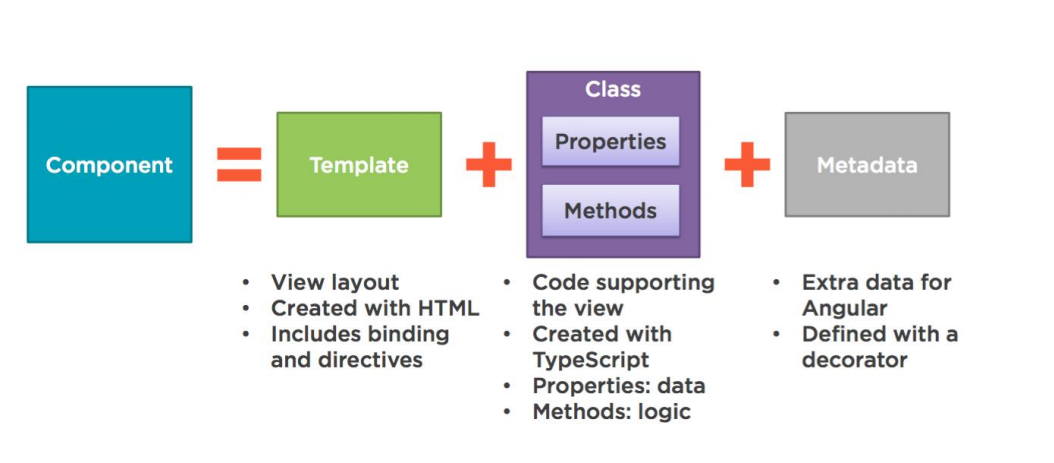
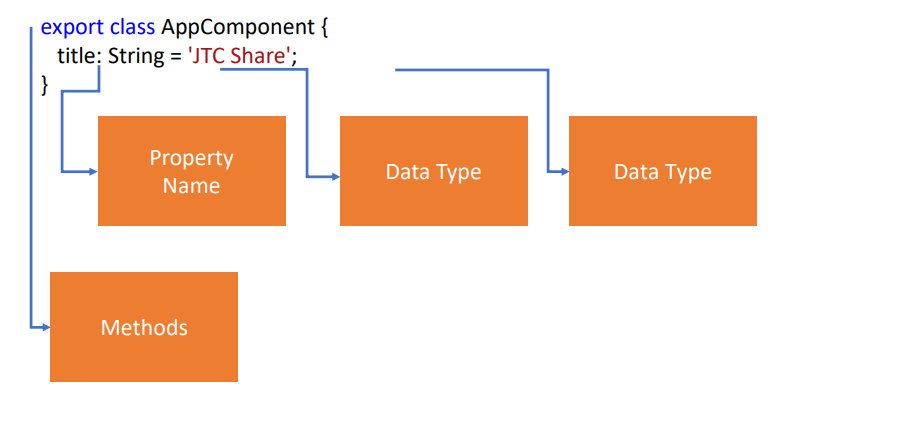
# Bài 3 Component trong Angular 10

# Khái niệm Component

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.

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”.  
[](https://laptrinhvien.io/wp-content/uploads/2018/01/laptrinhvienio-class.png)

Từ khoá “export” trước class làm cho component đó có thể được sử dụng bởi các component khác của ứng dụng. Và như chúng ta đã được học từ chương trước, file chứa mã bây giờ đã là một ES module. Nó sẽ được tải bởi module loader khác mà không cần thêm một script tag nữa.

Bên trong phần thân của class có những thuộc tính và phương thức.

**Định nghĩa Metadata với một Decorator**

* Một hàm thêm metadata vào một class, các thành phần thêm vào là những đối số nhất định.
* Tiền tố bắt đầu với @
* Là một trong những decorator được xây dựng sẵn bởi Angular: **@Component**

**Importing**

* Câu lệnh import cho phép chúng ta sử dụng những thành phần exported (xuất khẩu) từ module bên ngoài.
* Những module khác trong ứng dụng có thể import exported class nếu cần. Được phép sử dụng câu lệnh import xuyên suốt quá trình code để import bất kỳ thư viện bên thứ ba nào, hoặc những module của chúng ta đã viết, hay import từ chính Angular.
* Có thể import từ Angular, vì Angular cũng chính là một tập hợp những thư viện module. Mỗi thư viện là một module gồm nhiều tính năng liên quan. Khi chúng ta cần thứ gì đó từ Angular thì import một Angular module library (thư viện module). Giống như import bất kỳ module bên ngoài khác.

**Bootstrapping**

Hầu hết các ứng dụng Angular có một file index.html có chứa trang chính của ứng dụng. File index.html thường chỉ là một trang. Do đó một ứng dụng Angular thường được gọi là một Single Page Application (SPA).

Những điều tối thiểu phải có để tạo ra một AppModule

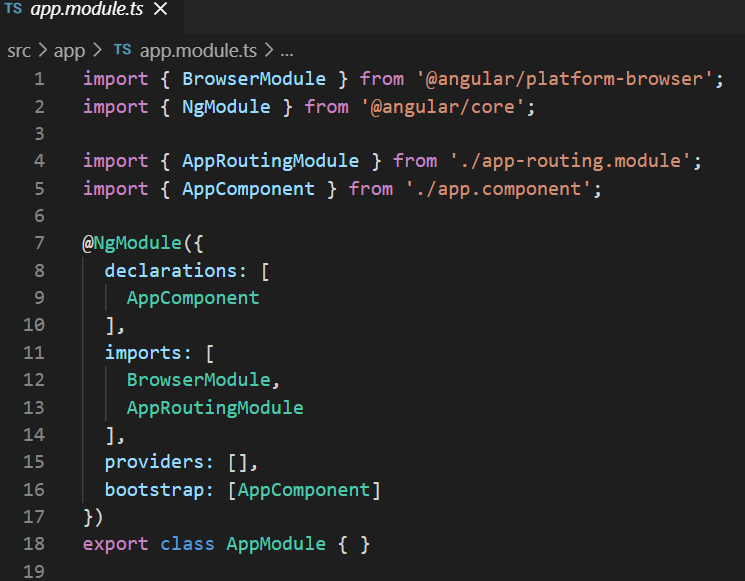
* @NgModule để xác định Class AppModule là một Angular Module.
* @NgModule cũng sẽ có những metadata object, cái mà sẽ nói cho Angular biết cách biên dịch và khởi chạy ứng dụng của chúng ta.
* Imports: import module BrowserModule, module này cần cho mọi ứng dụng chạy trên trình duyệt.
* Declarations: Khai báo những component sẽ được sử dụng trong module này.
* Boostrap: Component gốc được Angular tạo ra và chèn vào trong index.html

## Thành 1phần cha AppComponent

Trong bài trước, khi tạo project, chúng ta đã thấy các file được tạo ra trong thư mục src/app:

* **app.component.css**
* **app.component.html**
* **app.component.spec.ts**
* **app.component.ts**
* **app.module.ts**

Nếu bạn mở file **app.module.ts** ra, bạn sẽ thấy nội dung như sau:



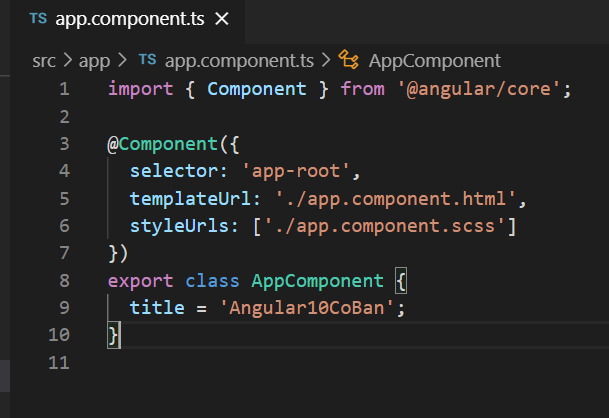
có tác dụng import các module và component cần thiết: BrowserModule, NgModule, AppComponent.

BrowserModule và NgModule là 2 module căn bản app nào cũng có. AppComponent là component được khai báo trong file **app.component.ts**

**Trong thẻ @NgModule có**

1. **declarations**: Dành cho những thứ bạn sử dụng trong các teamplates của mình: chủ yếu là các components (~views: các lớp hiển thị dữ liệu), bên cạnh đó cũng có các directives và pipes.
2. **Import**: Thêm các module từ trong thư viện Angular.
3. **provides**: Dành cho các services(~ models: các class nhận và xử lý dữ liệu).

Nội dung file app.component.ts:



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.

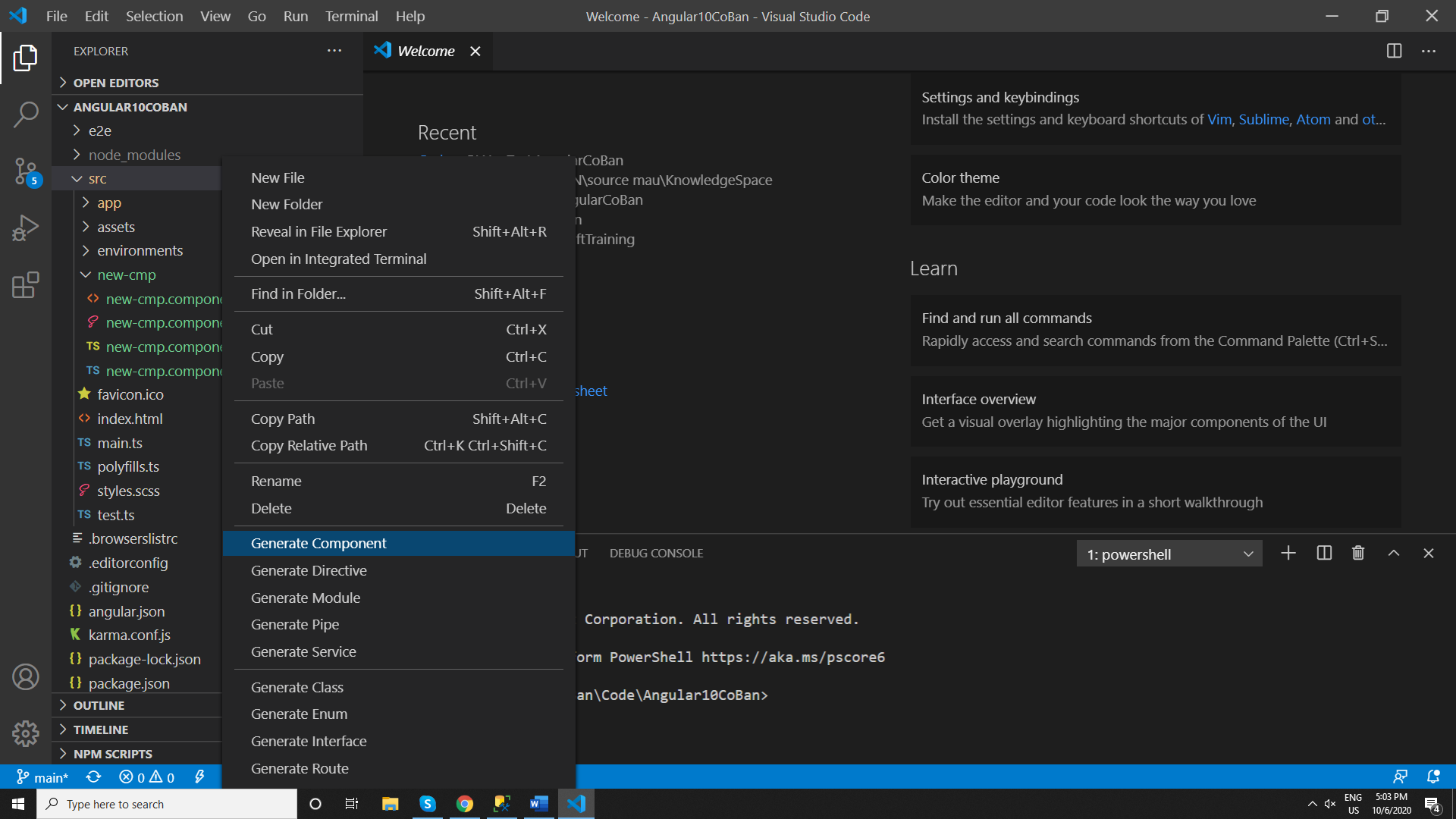
## 3. Tạo component

Để tạo component mới, ta sử dụng lệnh Angular CLI:

**ng g component <ten-component>**

Ví dụ, để tạo component có tên là new-cmp, ta chạy lệnh **ng g component new-cmp**

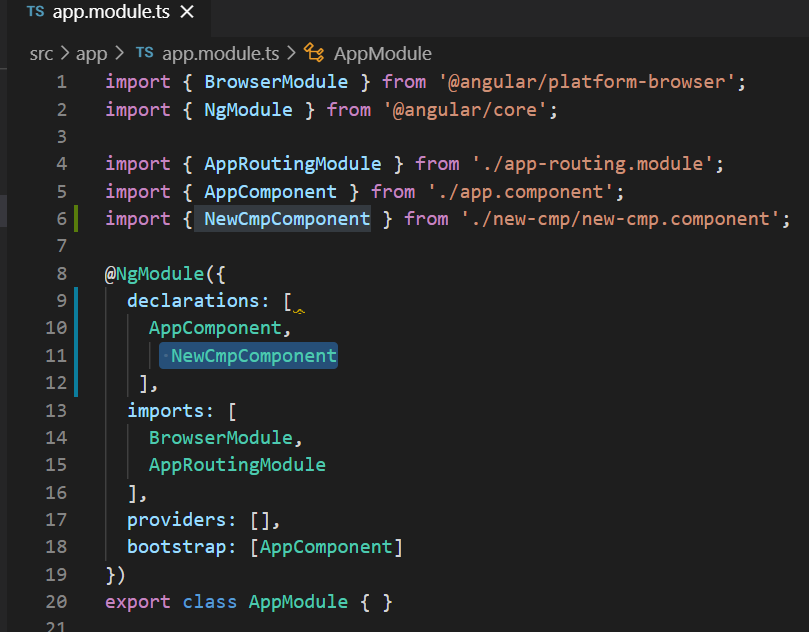
Nếu bạn đã cài extension Angular File thì chỉ cần chuột phải vào file và chon Gererate Component



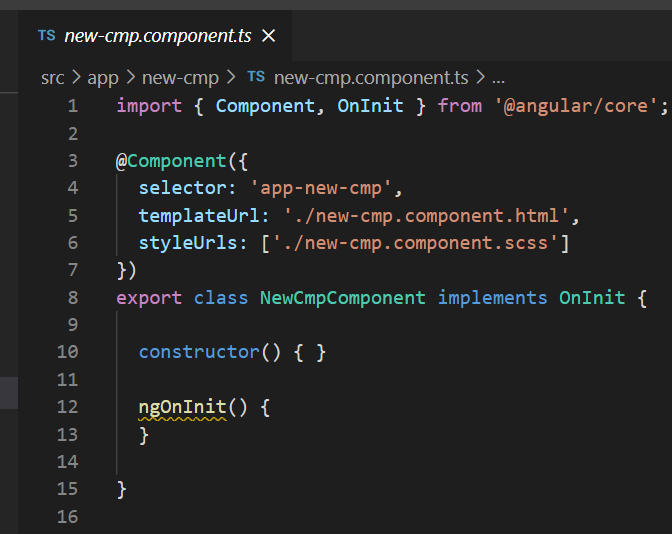
Vào kiểm tra cấu trúc thư mục thì ta thấy thư mục new-cmp được tạo ra trong thư mục **src/app/.** Trong thư mục **new-cmp** có 4 file được tạo ra::

* **new-cmp.component.css** − file css của component.
* **new-cmp.component.html** − file html của component.
* **new-cmp.component.spec.ts** − file sử dụng cho unit test.
* **new-cmp.component.ts** − file định nghĩa các module, thuộc tính

Ngoài việc tạo ra mới 4 file này, angular CLI cũng giúp chúng ta khai báo new component này cho project trong file app.module.ts



## Cấu trúc của component



Ở đây khai báo 1 class gọi là NewCmpComponent. Class này implements class OnInit. Class **OnInit** có constructor là ngOnInit(), nên trong NewCmpComponent cũng sẽ override lại constructor ngOnInit() này. Constructor ngOnInit() sẽ được gọi mặc định khi class được gọi đến.

Ở bài trước, ta đã tìm hiểu được luồng hoạt động của AppComponent: file **main.ts** được chạy đầu tiên. Trong file **main.ts** khai báo AppModule. AppModule khai báo sử dụng AppComponent. Trong App Component có selector <app-root>, <app-root> được gọi trong file index.html và cuối cùng file index.html được hiển thị lên trình duyệt.

Cách hoạt động của component khác cũng tương tự như App Component. Trong ví dụ trên, NewCmpComponent khai báo selector là <app-new-cmp>. Như vậy thẻ <app-new-cmp> sẽ "đại diện" cho NewCmpComponent, khai báo <app-new-cmp> ở đâu thì nội dung file html của NewCmpComponent sẽ được hiển thị ở chỗ đó, giống như khi khai báo <app-root></app-root> ở đâu thì nội dung file app.component.html được hiển thị ở đó.