

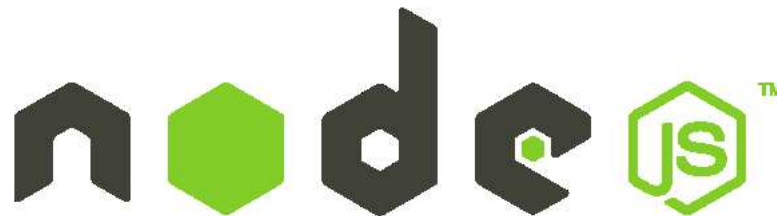
# Partie 2: Etape1

## Développement Front End

---

- HTML, CSS, Bootstrap, JS, JQUERY, JSON pour développer la partie Front End,
- L'utilisation de node js
- L'utilisation de Angular comme framework de développement,
- L'utilisation de Firebase pour hébergement de la partie front end,

# Le Framework Angular (1)



Ressources:

- <https://openclassrooms.com/fr/courses/4668271-developpez-des-applications-web-avec-angular>
- <https://angular.io/cli>
- <https://angular.io/tutorial>

# Le Framework Angular (2)

- Angular est une plateforme de développement qui permet de créer des applications web dynamiques et immersives.
- Angular est géré par Google;
- Le TypeScript: ce langage permet un développement beaucoup plus stable, rapide et facile.
- le framework permet le développement d'applications mobiles multi-plateformes à partir d'une seule base de code,

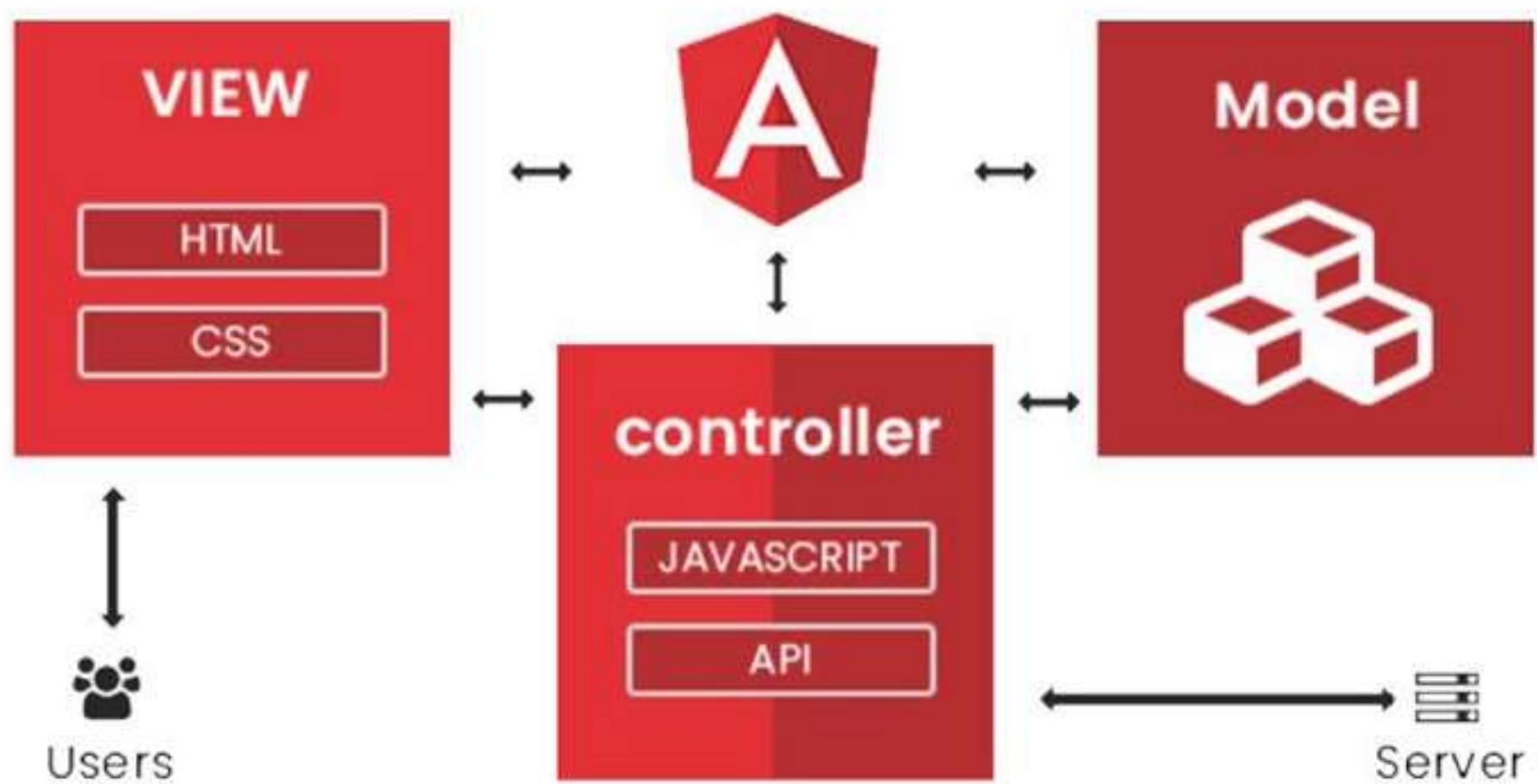
# Le Framework Angular (3)

- AngularJS est un framework de création d'applications Web SPA (single page applications). Les applications à page unique ou les SPA chargent tout le contenu d'un site sur une seule page. Cette page unique est généralement un fichier index.html.
- Cela signifie qu'une fois la page principale chargée, cliquer sur les liens ne rechargera pas la page entière mais mettra simplement à jour les sections de la page elle-même.

# MVC in Angular

- L'architecture MVC est essentiellement implémentée chaque fois que vous souhaitez former une application basée sur l'interface utilisateur.
- Avec MVC nous nous concentrerons sur la conception de votre interface utilisateur et sur la manière d'intégrer les données dans votre interface utilisateur.
- Vous allez diviser l'application complète en trois parties principales interconnectées : modèle, vue et contrôleur.

# MVC Architecture



# MVC:

## La couche Model (modèle)

- Model: Permettant de se connecter aux ressource de données, et de gérer les données de l'application. Après avoir représenter les données par des classes (Typescript) , vous pouvez implémenter le CRUD nécessaire sur la base. Il s'agit des fichiers services Angular (app.service.ts)

# MVC:

## La couche View (Vue)

- View: ou la couche Templates représentés par les fichiers html et css. Elle représente la partie UI, Reçoit les données du modèle pour leurs affichage final via le contrôleur, et vice versa (app.component.html & app.component.css)



# MVC:

## La couche Controller (Contrôle)

- Controller: Le contrôleur est chargé de contrôler l'interaction entre le modèle et la vue. Accepte les entrées et les convertit en commandes pour le modèle ou la vue. L'utilisateur lance l'exécution. Cet utilisateur enverra une demande au contrôleur, qui sera acceptée par le contrôleur. Ensuite, selon la demande, il rassemble les données et manipule le modèle. Maintenant, la vue reçoit les mêmes données que celles que le modèle écrit. La vue est juste un modèle HTML représentant la vue dans Angular (app.component.ts)

# Le Framework Angular (4)

- **Prérequis :**
  - Des connaissances solides en HTML
  - Des bases en TypeScript ou une bonne connaissance du JavaScript
  - Des bases en CSS/SCSS
- **Outils nécessaires :**
  - un éditeur de code (VS Code, sublime, ...) ;
  - Node.js ;
  - NPM ;
  - Angular CLI.

# Préparation de l'Environnement de développement

- Toutes les installations sur l'invite de commande cmd ou Git Bash,
    - 1- Installer Node.JS: <https://nodejs.org/en/>
    - 2- Vérifier quelle version:  
Node - version
    - 3- Installer NPM:  
`npm install -g npm@latest`
      - Npm est un package manager qui permet l'installation d'énormément d'outils et de bibliothèques dont vous aurez besoin pour tout type de développement. Pour l'installer, ouvrez une ligne de commande et tapez les commandes du lien précédent,
    - 4- Installer Angular CLI:  
`npm install -g @angular/cli`  
<https://angular.io/cli#installing-angular-cli>
-

# Préparation de l'Environnement de développement

- **Qu'est-ce que le CLI ?**
  - Le CLI, ou “Command Line Interface” (un outil permettant d'exécuter des commandes depuis la console), d'Angular est l'outil qui vous permet d'exécuter des scripts pour la création, la structuration et la production d'une application Angular.
- Pour plus d'informations:
  - node.js : <https://nodejs.org/en/docs/>
  - npm : <https://docs.npmjs.com/>
  - ng : <https://cli.angular.io/>

# Nouveau projet Angular

Sur l'invite de commandes, taper la commande:

```
>> ng new mon-projet
```

opération qui prend quelques minutes pour la création des composantes de base de l'application. Il y aura aussi le choix du langage de styles (css) et une question sur l'utilisation des routes (il faut répondre par 'y').

sous le dossier mon-projet vous trouverez l'arborescence suivante:

# Nouveau projet Angular

Nom	Modifié le	Type	Taille
e2e	28/02/2021 11:49	Dossier de fichiers	
node_modules		Dossier de fichiers	
src		Dossier de fichiers	
.browserslistrc	28/02/2021 11:49	Fichier BROWSERS...	1 Ko
.editorconfig	28/02/2021 11:49	Fichier EDITORCO...	1 Ko
.gitignore	28/02/2021 11:49	Document texte	1 Ko
angular.json		Fichier JSON	4 Ko
karma.conf		Fichier JavaScript	2 Ko
package.json	28/02/2021 11:49	Fichier JSON	2 Ko
package-lock.json	28/02/2021 12:05	Fichier JSON	571 Ko
README.md	28/02/2021 11:49	Fichier MD	1 Ko
tsconfig.app.json	28/02/2021 11:49	Fichier JSON	1 Ko
tsconfig.json	28/02/2021 11:49	Fichier JSON	1 Ko
tsconfig.spec.json	28/02/2021 11:49	Fichier JSON	1 Ko
tslint.json	28/02/2021 11:49	Fichier JSON	4 Ko

Les fichiers source

Le fichier de configuration

# Nouveau projet Angular

**1) Toujours sur cmd ou depuis votre terminal  
vscode; compiler le projet** et démarrer le serveur de  
développement, la commande ouvre le résultat sur:  
`http://localhost:4200`

*>> ng serve - o*

# Nouveau projet Angular

## **Vous pouvez Installer Bootstrap:**

```
>> npm install bootstrap@3.3.7 --save
```

- Cette commande téléchargera Bootstrap et l'intégrera dans le package.json du projet. Il vous reste une dernière étape pour l'intégrer à votre projet. Ouvrez le fichier angular.json du dossier source de votre projet. Dans "architect/build/options", modifiez le tableau styles comme suit :

```
"styles": [  
  "../node_modules/bootstrap/dist/css/bootstrap.css",  
  "styles.scss"  
],
```



# Nouveau Projet Angular

- A la fin des étapes de création du nouveau projet, l'arborescence suivante est créée:

## FOLDERS

- ▼ mon-projet
  - ▶ e2e
  - ▶ node\_modules
  - ▼ src
    - ▶ app
    - ▶ assets
    - ▶ environments
    - favicon.ico
    - <> index.html
    - main.ts
    - polyfills.ts
    - /\* styles.css
    - test.ts
- .browserslistrc
- .editorconfig
- .gitignore
- /\* angular.json
- /\* karma.conf.js
- /\* package-lock.json
- /\* package.json
- <> README.md
- /\* tsconfig.app.json
- /\* tsconfig.json
- /\* tsconfig.spec.json
- /\* tslint.json

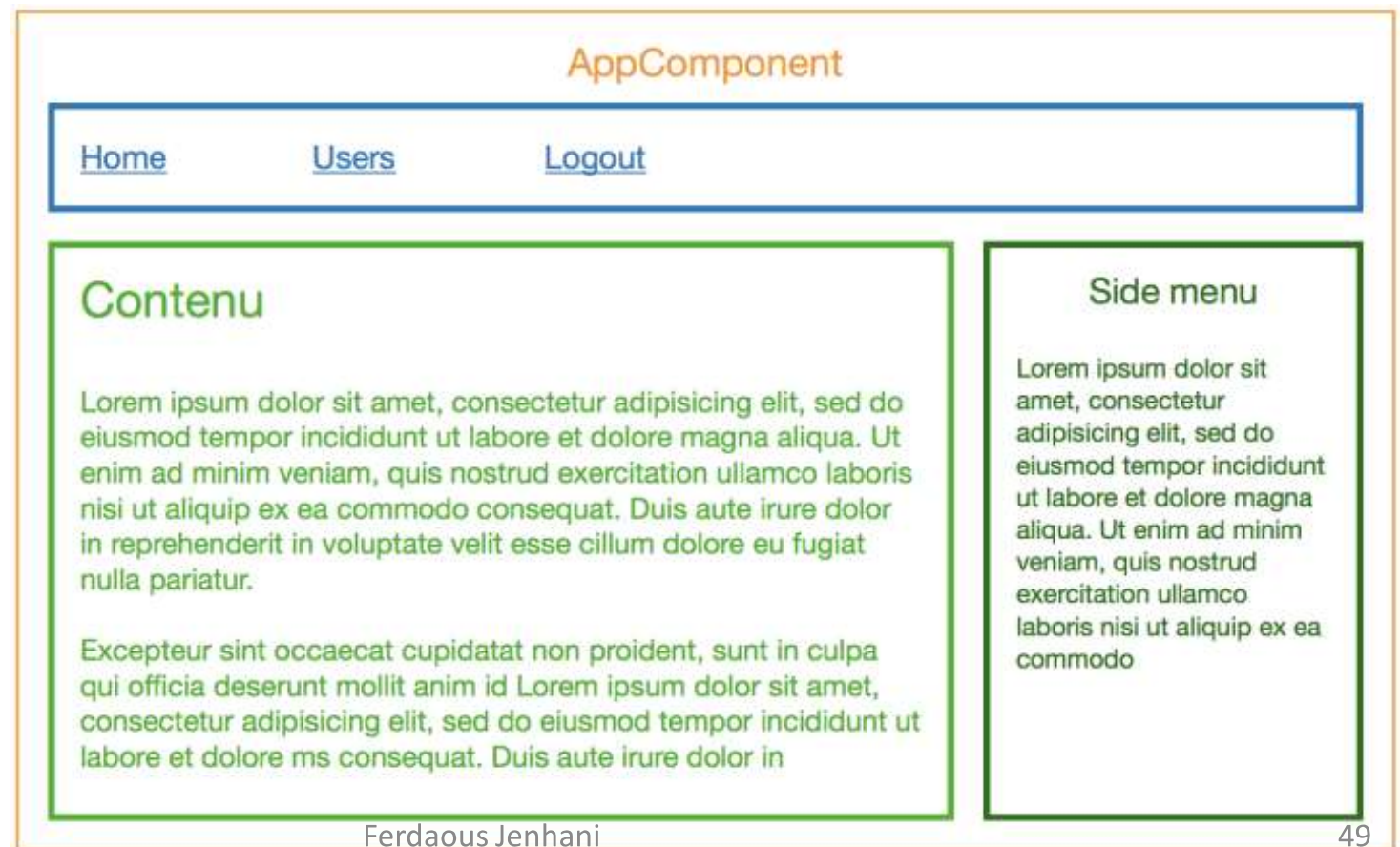
```
index.html
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>MonProjet</title>
6   <base href="/">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link rel="icon" type="image/x-icon" href="favicon.ico">
9 </head>
10 <body>
11   <app-root></app-root>
12 </body>
13 </html>
14
```

## FOLDERS

- ▼ mon-projet
  - ▶ e2e
  - ▶ node\_modules
  - ▼ src
    - ▼ app
      - app-routing.module.ts
      - /\* app.component.css
      - <> app.component.html
      - app.component.spec.ts
      - app.component.ts
      - app.module.ts

# Components Angular

- Un projet Angular est structuré autour de composantes. App est la composante de base et la racine de l'application et créée automatiquement.



# Components Angular

- Les components sont les composantes de base d'une application Angular, une application est une arborescence de components,
- Le composant AppComponent est la composante de base d'une application, tous le reste des components y seront emboîtés. Elle est créée automatiquement à la création du projet,
- On peut avoir un component de chaque élément d'une page web, ou de chaque page web,

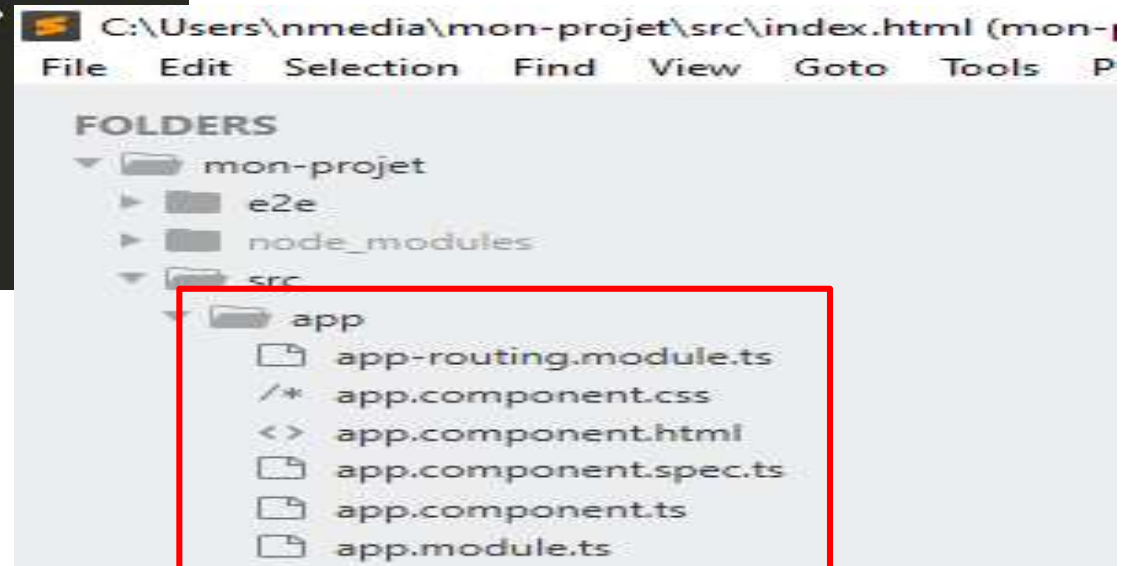
# App Component

- Le component « app » est créé automatiquement.

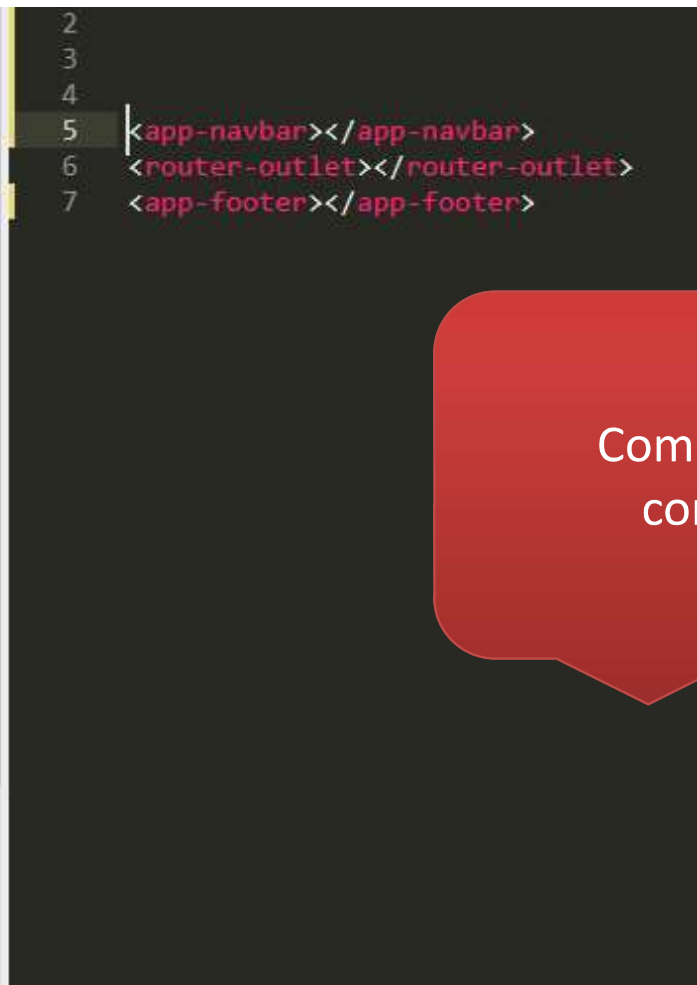
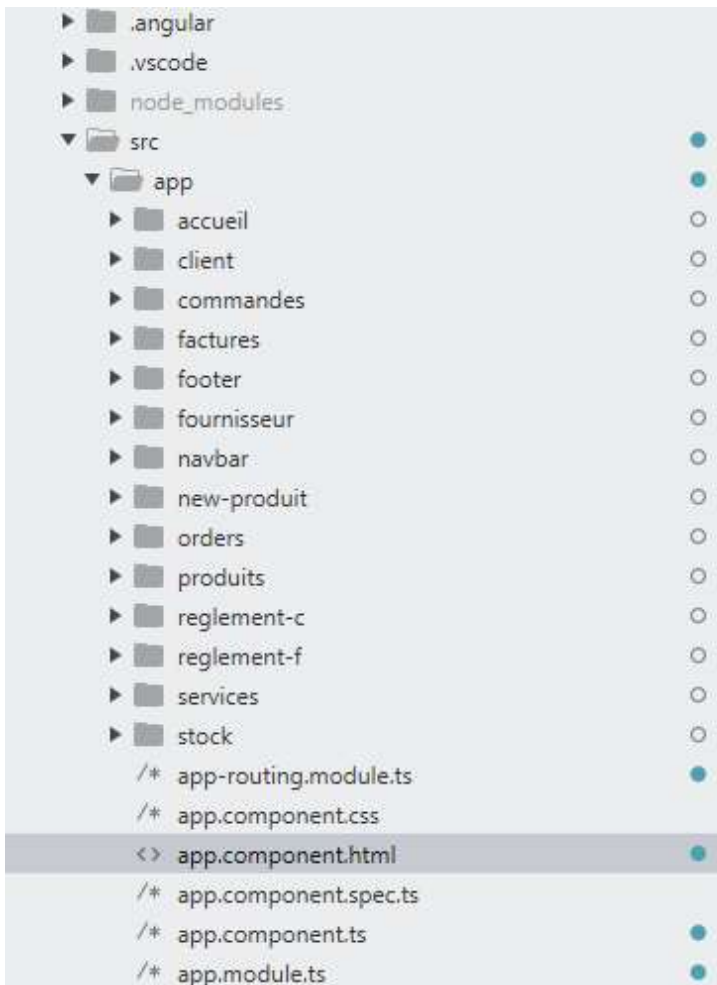
Text (UNREGISTERED)

ces Help

```
index.html x
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>MonProjet</title>
6   <base href="/">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link rel="icon" type="image/x-icon" href="favicon.ico">
9 </head>
10 <body>
11   <app-root></app-root>
12 </body>
13 </html>
14
```



- Effacer le contenu du fichier **app.component.html** et garder uniquement le selecteur `<router-outlet></router-outlet>`
- Créer les composants «navbar » et « footer » et insérer dans le fichier « **app.component.html** »



Comment Créer un  
component???



# Créer un component

```
>> cd C:/Users/mon_projet
```

```
>> ng generate component mon-premier
```

```
installing component
  create src/app/mon-premier/mon-premier.component.scss
  create src/app/mon-premier/mon-premier.component.html
  create src/app/mon-premier/mon-premier.component.spec.ts
  create src/app/mon-premier/mon-premier.component.ts
  update src/app/app.module.ts
```

→ Le fichier app.module.ts sera modifié

**FOLDERS**

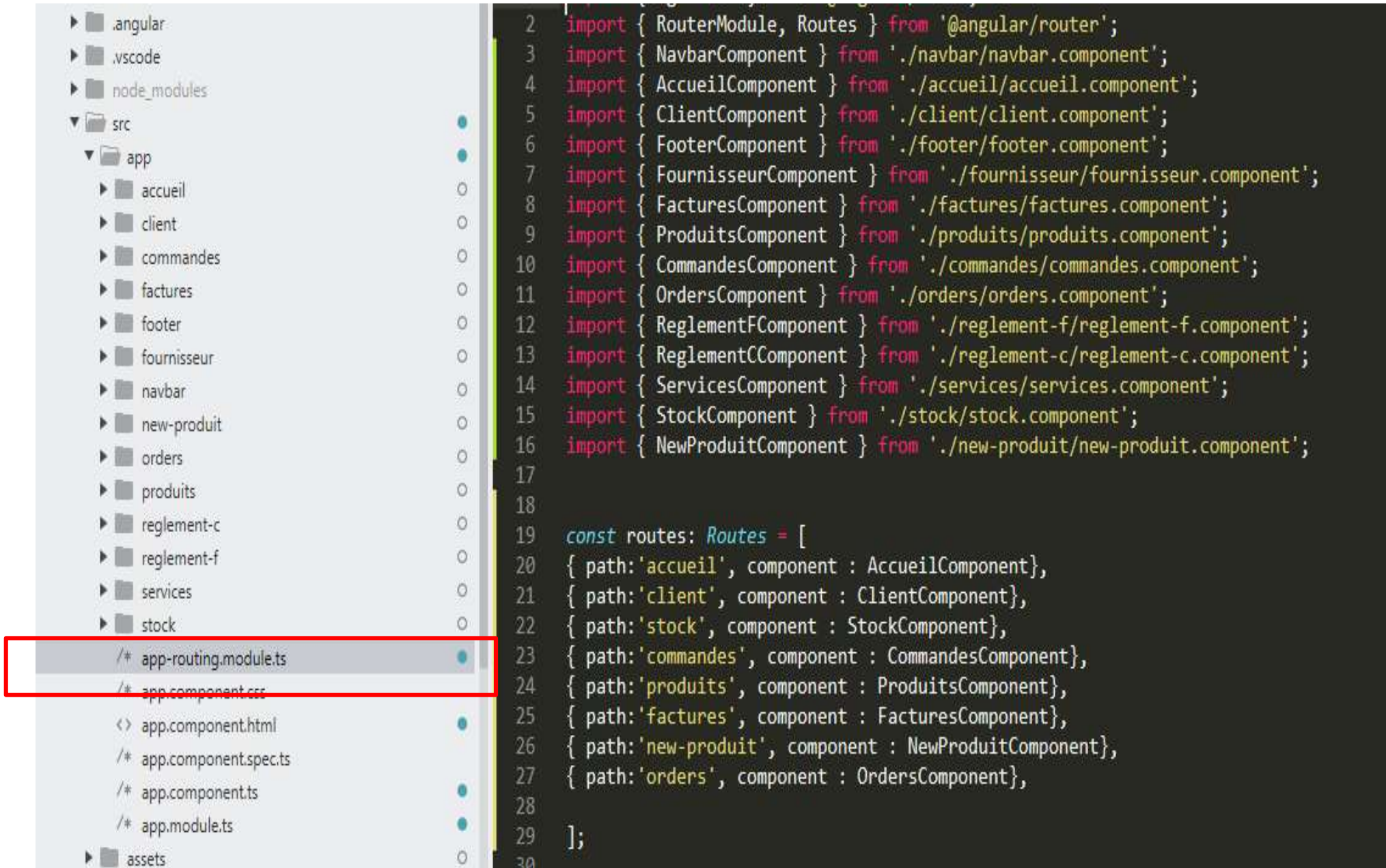
- mon-projet
  - e2e
  - node\_modules
  - src
    - app
      - mon-premier**
        - mon-premier.component.css
        - mon-premier.component.html
        - mon-premier.component.spec.ts
        - mon-premier.component.ts
      - app-routing.module.ts
      - app.component.css
      - app.component.html
      - app.component.spec.ts
      - app.component.ts
      - app.module.ts**
    - assets
    - environments
    - favicon.ico

```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModuleModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { MonPremierComponent } from './mon-premier/mon-premier.component';
7
8 @NgModule({
9   declarations: [
10     AppComponent,
11     MonPremierComponent
12   ],
13   imports: [
14     BrowserModule,
15     AppRoutingModuleModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
20 export class AppModule { }
21
```

À la création d'un nouveau component, Ces imports sont insérés automatiquement dans le fichier app.module.ts



# Angular: Routes et Navigation



The image displays the file structure of an Angular application on the left and the corresponding routing module code on the right.

**File Structure (Left):**

- angular
- .vscode
- node\_modules
- src
  - app
    - accueil
    - client
    - commandes
    - factures
    - footer
    - fournisseur
    - navbar
    - new-produit
    - orders
    - produits
    - reglement-c
    - reglement-f
    - services
    - stock
    - /\* app-routing.module.ts**
    - /\* app.component.css**
    - app.component.html
    - /\* app.component.spec.ts
    - /\* app.component.ts
    - /\* app.module.ts
  - assets

**Routing Module Code (Right):**

```
2 import { RouterModule, Routes } from '@angular/router';
3 import { NavbarComponent } from './navbar/navbar.component';
4 import { AccueilComponent } from './accueil/accueil.component';
5 import { ClientComponent } from './client/client.component';
6 import { FooterComponent } from './footer/footer.component';
7 import { FournisseurComponent } from './fournisseur/fournisseur.component';
8 import { FacturesComponent } from './factures/factures.component';
9 import { ProduitsComponent } from './produits/produits.component';
10 import { CommandesComponent } from './commandes/commandes.component';
11 import { OrdersComponent } from './orders/orders.component';
12 import { ReglementFComponent } from './reglement-f/reglement-f.component';
13 import { ReglementCComponent } from './reglement-c/reglement-c.component';
14 import { ServicesComponent } from './services/services.component';
15 import { StockComponent } from './stock/stock.component';
16 import { NewProduitComponent } from './new-produit/new-produit.component';
17
18
19 const routes: Routes = [
20   { path: 'accueil', component: AccueilComponent },
21   { path: 'client', component: ClientComponent },
22   { path: 'stock', component: StockComponent },
23   { path: 'commandes', component: CommandesComponent },
24   { path: 'produits', component: ProduitsComponent },
25   { path: 'factures', component: FacturesComponent },
26   { path: 'new-produit', component: NewProduitComponent },
27   { path: 'orders', component: OrdersComponent },
28
29 ];
30
```

# Angular: Routes et Navigation

- Le fichier **app-routing-module.ts** est créé automatiquement lors de la création d'une application et avec le component de base app,
- Il contient les différents liens de navigation entre les pages créées,
- Angular CLI importe automatiquement RouterModule et Routes,