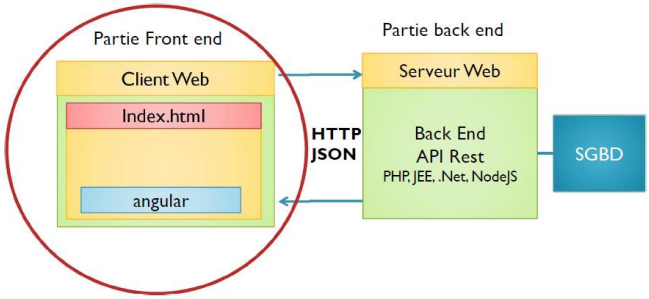
TP 1 : Configuration de l'environnement de développement- Notions théoriques

I- Introduction

- Angular est un Framework de developpement Javascript. Il permet de créer des applications client (Client-side application) en utilisant HTML, CSS et Javascript (TypeScript)
- Angular permet de créer des applications Web basées sur une seule page (Single Page Application)
- Une SPA est une application qui contient une seule page HTML (index.html) récupérée du serveur.



II- Historiques :

- **Angular 1 (Agular JS) :**
  - Première version de Angular qui est la plus populaire.
  - Elle est basée sur une architecture MVC coté client. Les applications Angular 1 sont écrite en Java Script.
- **Angular 2 (Angular) :**
  - Est une réécriture de Angular 1 qui est plus performante, mieux structurée et représente le futur de Angular.
  - Les applications de Angular2 sont écrites en Type Script qui est compilé et traduit en Java Script avant d'être exécuté par les Browsers Web.Angular 2 est basée sur une programmation basée sur les Composants Web (Web Component)
- **Angular 4, 5, 6,7,8,… :** sont de simples mises à jour de Angular 2 avec des améliorations au niveau performances.

III- Préparation de l'environnement

- Installation de Node Js (www.NodeJs.org)
- Installation de Angular CLI : *npm install -g @angular/cli*
- Installation d'un IDE : Visual Studio code

Commandes CLI à utiliser fréquemment

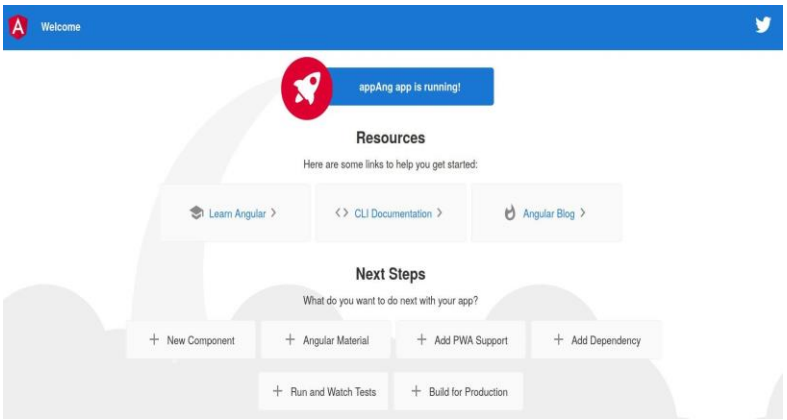
Commandes	Besoin
ng new firstApp	Créer un projet angular
ng serve	Exécuter un projet
ng g component componentName	Créer un composant
ng g service serviceName	Créer un service
ng g class className	Créer une classe

IV- Activité :

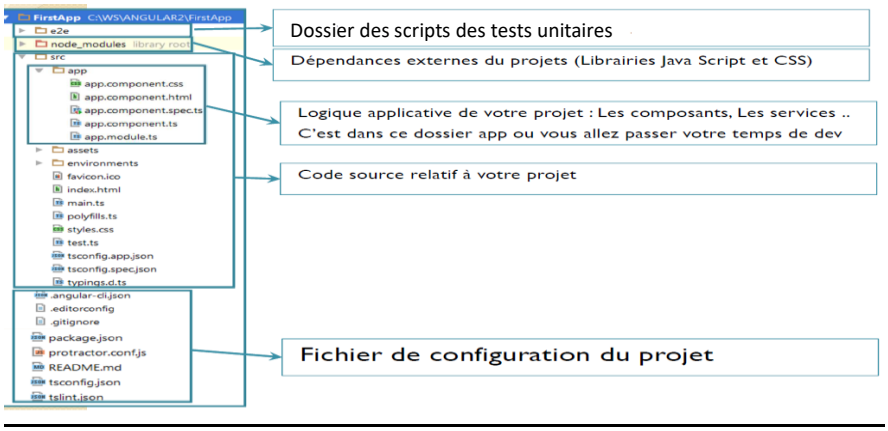
1- Créer un nouveau projet Angular : *ng new FirstApp*

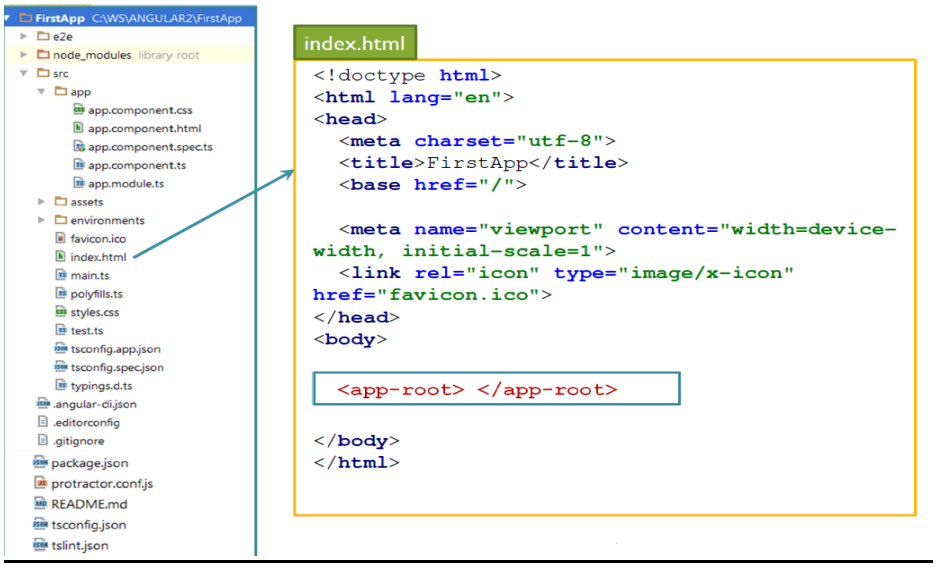
2- Lancer ce projet: *ng serve*

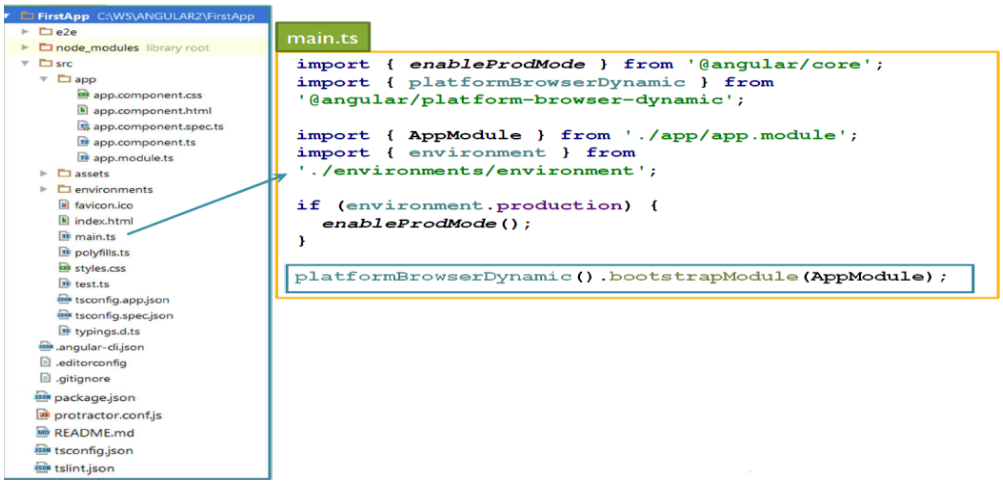
3- Ouvrir le navigateur et taper l'url : *http:// localhost:4200*

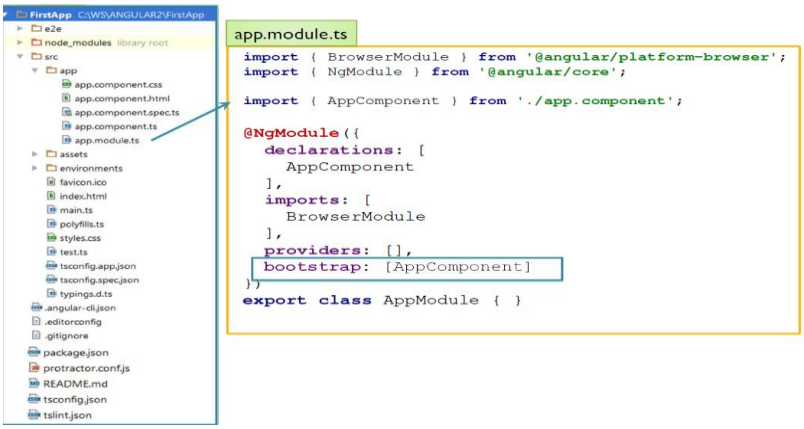


V- Structure du Projet Angular





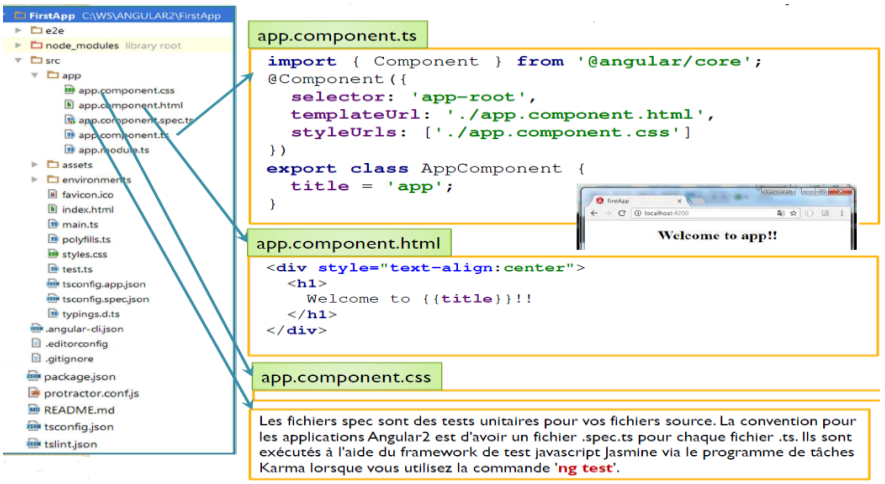




Declarations : C'est le tableau de composants. Si un nouveau composant est créé, il sera importé en premier et la référence sera incluse dans les déclarations

Imports : C'est un tableau de modules requis pour être utilisé dans l'application providers**.** Cela va contenir tous les services crées

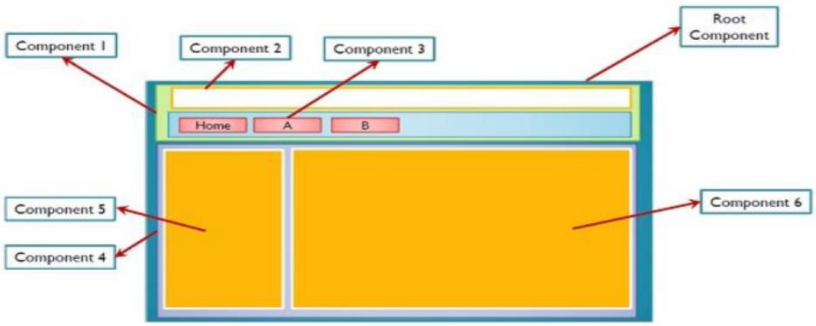
Bootstrap : Cela inclut le composant principal de l'application pour démarrer l'exécution.



VI- Componenent

Chaque composant se compose principalement des éléments suivants :

- HTML Template : représentant sa vue
- Une classe représentant sa logique métier
- Une feuille de style CSS
- Un fichier spec sont des tests



Un composant peut être inséré dans n'importe quelle partie HTML de l'application en utilisant son sélecteur associé.

@Component

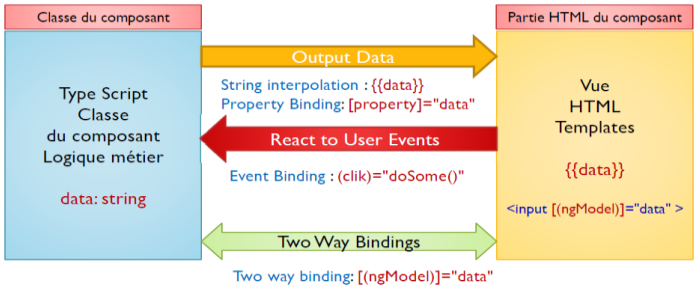
Un composant est une classe qui possède le décorateur ***@Component***. Ce décorateur possède les propriétés suivantes :

- selector : permet de spécifier le tag (nom de la balise) associé à ce composant.
- templateUrl : permet d'associer un fichier externe HTML contenant la structure de la vue du composant
- styleUrls : spécifier les feuilles de styles CSS associées à ce composant

VII- DataBinding

Pour insérer dynamiquement des données de l'applications dans les vues des composants, Angular définit des techniques pour assurer la liaison des données.

Data Binding = Communication



String interpolation : est une technique de One Way Binding, elle utilise l'expression {{ }} pour afficher les données du composant dans la vue.

Exemple :

```
export class AppComponent {
  title = 'TP 1 Angular';

}
```

app.component.ts

```
<h2>{{title}}</h2>
```

app.component.html

Property Binding : est une autre technique One Way Binding. Elle permet de lier une propriété de la vue avec une propriété définie dans le composant.

Exemple :

```
urlImg="./assets/images/pc portable.jfif"
```

```
<img [src]=urlImg>
```

Event Binding : dans Angular , event binding est utilisé pour gérer les événements déclenchés comme le clic de bouton, le déplacement de la souris, etc. Lorsque l'événement se produit, il appelle la méthode spécifiée dans le composant.

```
afficher()
{
  console.log("hello");
}
```

```
<button (click)="afficher()">Afficher</button>
```

Two-way binding : nous avons vu que dans la le One Way Binding, tout changement dans la vue n'était pas reflété dans le composant. Pour résoudre ce problème, Angular Two Way Binding.

```
texte:string="hello";
```

```
<input type="text" [(ngModel)]=texte> {{texte}}
```

Remarque : il faut ajouter « ***FormsModule***» dans le tableau ***imports*** du fichier ***app.module.ts***