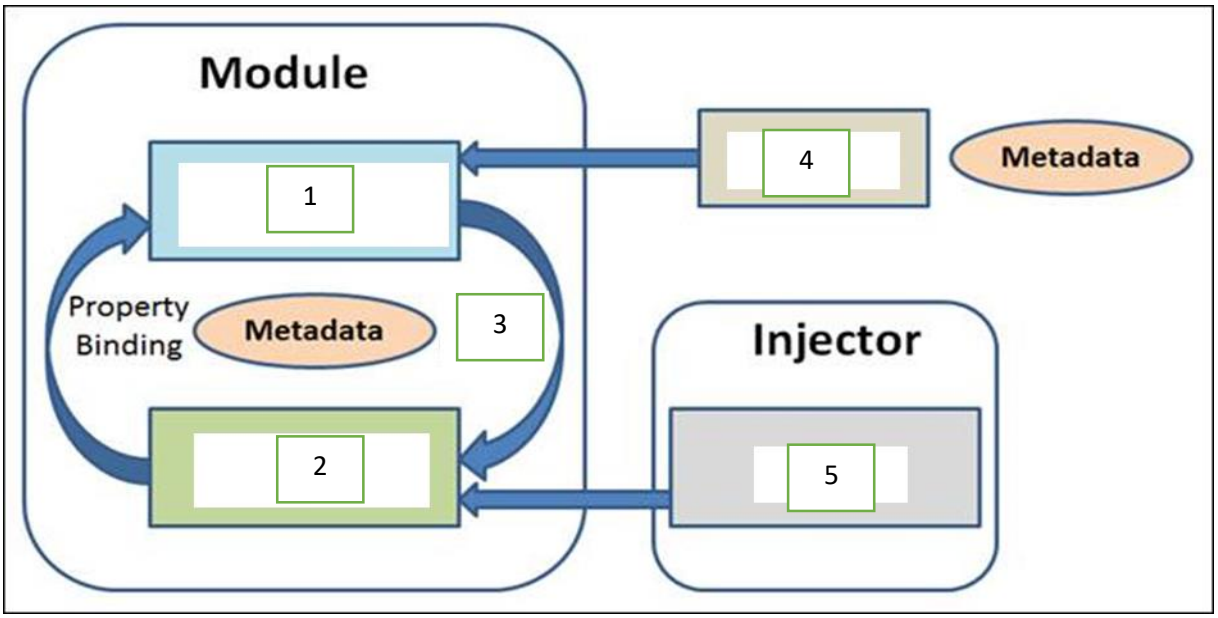
esprit Se former autrement	Année Universitaire : 2020-2021 ANGULAR

Révision

Partie 1

<u>Répondez aux questions suivantes sur la feuille de réponses fournie avec l'énoncée</u>

1- Complétez la figure suivante par les termes qui manquent. **(2 pts)**



2- Quels sont les différents types de directives dans Angular ? Citez un exemple de directive pour chaque type. **(1 pt)**
3- C'est quoi le rôle de la directive NgModel ? **(1 pt)**
4- Que permet de faire la déclaration suivante : providedIn : 'CustomersModule'? **(1 pt)**
5- C'est quoi un décorateur ? Citez un décorateur de classe et un décorateur de propriété. **(1 pt)**
6- Citez 3 moyens de communication entre deux composants? **(1 pt)**

Partie 2 : ***<u>Répondez aux questions suivantes :</u>***

Année Universitaire : 2020-2021

Révision

1 Écrivez le code qui vous permet d'afficher autant de composants todo <app-todo> autant d'éléments dans le tableau listTodo? **(1pt)**

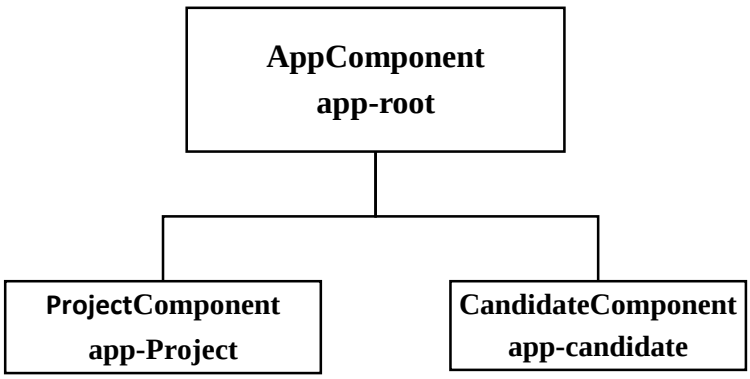
2 Pourquoi utilise-t-on le mot “export” dans une classe Typescript ? (**1pt**)

3 Quelle syntaxe utilisez-vous pour lier le click sur un bouton dans votre Template et une méthode getColor() dans le component.ts? (**1pt**)

4 Pour déclarer un module on utilise le décorateur @ngModule, citez l'utilité des paramètres ‘declarations’ et ‘imports’? (**2pts**)

Partie 3 :

Nous allons développer la partie front-end d’une application web pour les projets freelances avec Angular 9. Dans cette application un développeur freelance peut postuler à un projet. Pour cela, nous proposons la hiérarchie de composants suivante que nous allons créer tout au long de notre étude de cas.



Soit les modèles de données « **Project** » et « **Profile** »:

```
export class Project {
   title : string;
   description: string;
   budget: number;
   available:boolean;
```

```
export class Profile {
   name : string;
   bib: string;
   technologies: Technology[];
   hourPrice:number;
```

Révision

}	}

Soit deux liens projects et Candidate au niveau du fichier app.component.html.

- En cliquant sur le lien « projects », un composant **ProjectComponent** est chargé dans le Template du composant AppComponent et qui affiche la liste des offres.
- En cliquant sur le lien « Candidate », un composant **CandidateComponent** est chargé dans le Template du composant AppComponent et qui affiche un formulaire pour postuler à une offre.

1- Complétez le code qui manque dans chaque numéro dans le contenu du Template **AppComponent**.

app.component.html (1 pt)

```
<a href="#">Home</a>
<a 1 ="/projects">Project</a>
<a 1 ="/candidate">Candidate</a>
< 2 ..... ></ ... 2 >
```

2- Au niveau du fichier app-routing.module.ts , configurez les routes qui permettent de pointer sur les composants ProjectComponent et CandidateComponent et dont les liens correspondants sont déjà définis dans AppComponent **(2 pts)**

```
const routes: Routes = [
1

];
@NgModule({
  declarations: [],
  imports: [
    CommonModule,
      2
  ],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

Révision

3- Soit la classe **ProjectService** suivante. La méthode getListProject() permet de retourner la liste des projets déclarés dans un fichier json. Complétez le code qui manque dans chaque numéro. **(1.5 pts)**

```
import { Injectable } from '@angular/core';
import { HttpClient } from [ 1 ];

import { Projet } from './model/Intership';
@Injectable({
  providedIn: 'root'
})
export class ProjetService {
  constructor(private _http : [ 2 ] ) { }
  getListProjet(){
    return this._http.get< [ 3 ] >("assets/Projet.json");
  }
}
```

4- Complétez par le code nécessaire pour la récupération de la liste des projets à partir de la méthode getListProjet() du service ProjetService. (**2 pts**)

```
import { Component, OnInit } from '@angular/core';
import { ProjetService } from '../Projet.service';
@Component({
  selector: 'app-projet',
  templateUrl: './projet.component.html',
  styleUrls: ['./projet.component.css']
})

export class ProjetComponent implements OnInit {
  [ 1 ]
  constructor( [ 2 ] ) { }

  ngOnInit(): void { [ 3 ] }
}
```

Révision

5- Complétez le code html suivant pour afficher seulement le titre et la description de chaque projet. Ajoutez le code nécessaire, au niveau du numéro 3, pour que la couleur d'arrière-plan du projet soit rouge si le projet n'est pas disponible et vert s'il est disponible **(2 pts)**

```
<div  1  ="let p of  2  "   3   >
          4
</div>
```

6- Au-dessus de la liste des projets, on va ajouter un champ de saisie permettant **à l'utilisateur de chercher un projet selon un budget**. Sachant qu'on a déclaré une propriété **budgetMax** de type number dans le composant ProjectComponent, complétez le code du Template de ce composant pour assurer la recherche. **(2 pts)**

```
<input type= 'number'  1  ="budgetMax"  >

<div …="let p of ">
   (Contenu déjà demandé dans la question 5)
<div  2  >

    (Contenu déjà demandé dans la question 5)
    </div>
</div>
```

****

Le composant **CandidateComponent** contient un formulaire pour qu'un développeur publie son profile.

7- Le champ « Name » du formulaire est **obligatoire**. Si ce champ est vide alors un message d'alerte s'affiche. Complétez alors le code HTML suivant : (**2 pts)**

```
<form #f="ngForm">
            <label>Name</label>
            <input type="text" name="nom" required   1   >
            <div   2   >
```

Révision

```
                The name is a required field
            </div>

</form>
```

8- Le bouton « Publish» du formulaire est **caché** si **le formulaire est invalide**. Mettez à jour le code html en ajoutant le code nécessaire au niveau du bouton « Publish » **(1 pt)**

```
<button type="submit"  *  > Publish</button>
```

9- En cliquant sur le bouton « Publish », le profile sera ajouté dans une liste des profiles définit dans le composant **CandidateComponent, list : Profile[].** Mettez à jour la classe du composant **CandidateComponent** en indiquant uniquement le code utile pour répondre à cette question. **(1,5 pts)**