**République Algérienne Démocratique et Populaire**

**Ministère de l'Enseignement Supérieur et de la Recherche Scientifique**

**Université Benyoucef BENKHEDDA- Alger1**

**Faculté des Sciences**

**Département Mathématiques et Informatique**

****

|  |
| --- |
| Rapport Test Web |

**Réalisé par**

Dahdouh Ahmed

Benbaba Rym Amina

2019/2020

Table des matières

[**Introduction** 3](#_Toc31924241)

[**La partie back-end** 3](#_Toc31924242)

[**La partie front-end** 5](#_Toc31924243)

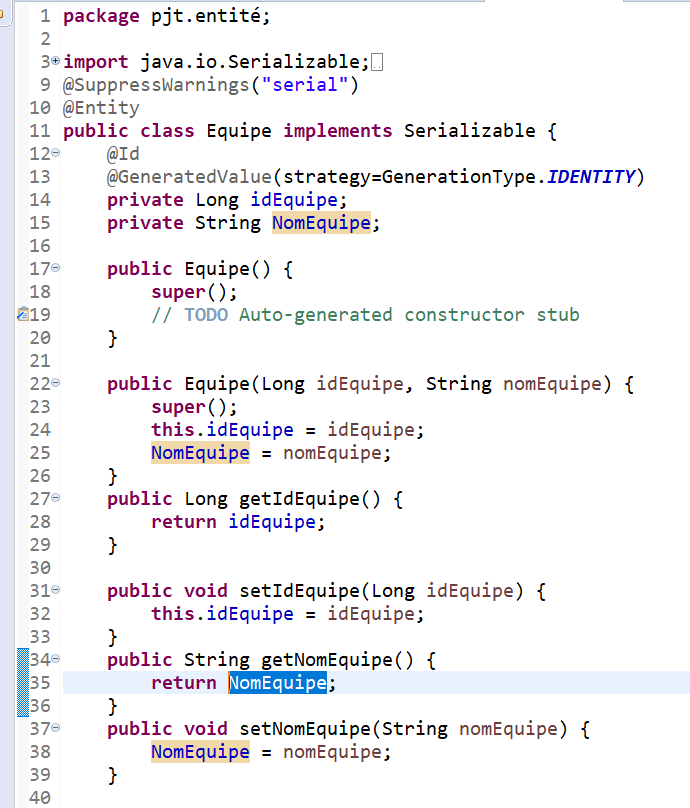
[**Interfaces** 12](#_Toc31924244)

# **Introduction**

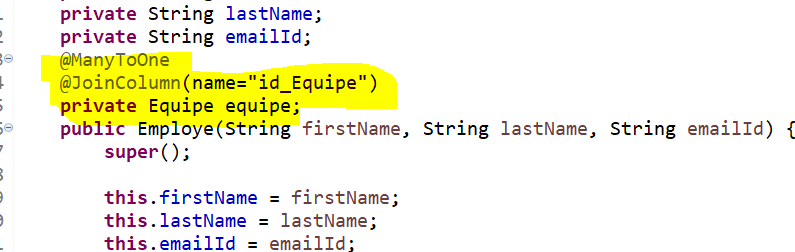
Afin de réaliser ce TP et bien organisé les étapes de travail, nous allons travailler sur les deux parties : La partie Back-End et la partie Front-end.

# **La partie back-end**

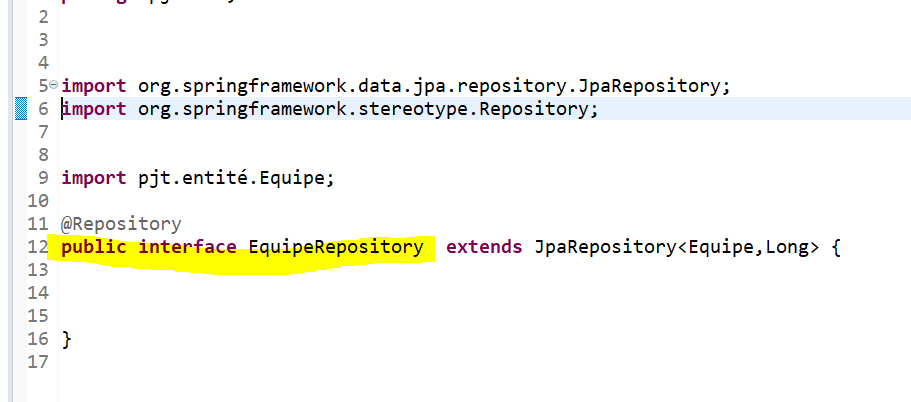
La première étape consiste à définir la class Equipe comme le montre la figure suivante :



Après cela nous avons modifier la class Employée par l’ajout des dépendances @ManyToOne afin de générer la clé étrangère comme suit :



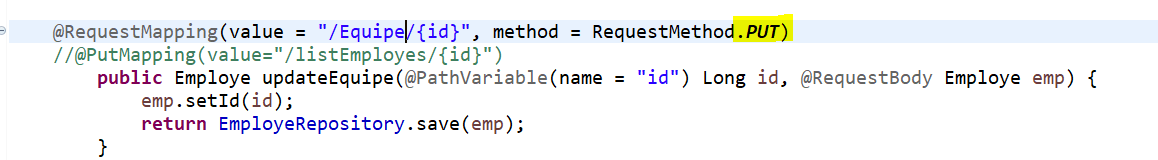
Dans la couche DAO vous avons créé l’interface suivante :



Et enfin dans la couche service nous avons définit deux fonctions, La première fonction consiste a créé des employées avec leurs équipes attribués, cette fonction prend en paramètres deux attributs qui sont (le nombre des employées et le nombres des équipes) par la suite une variable affectation qui prend la valeur de nombre des employées divisé sur le nombre des équipes (pour définir le nombre de employées qui sont dans la même équipe), donc nous avons deux boucle, une pour le nombre d’équipes et l’autre pour l’affectation et tous cela pour crée des nouveaux employées et les enregistrer dans la BDD avec leurs équipes :



La deuxième fonction consiste à mettre a jours l’équipe d’un employée, donc c’est une simple fonction qui prendre en paramètre le id de l’employée et change son équipe et enfin le sauvegarder.



Il est à noté que nous avons utilisé la fonction deleteAll() pour supprimer le contenu de la base à chaque appel de la fonction d’affectation.

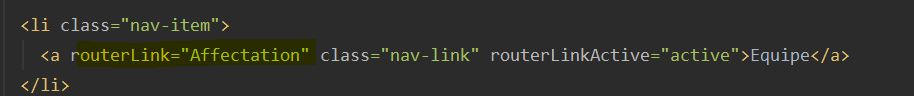


# **La partie front-end**

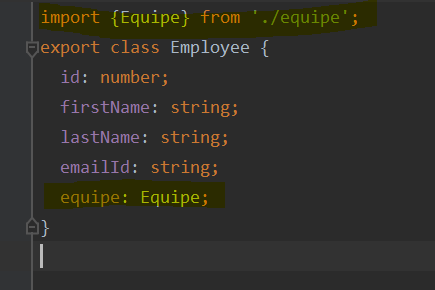
Pour la partie front end nous avons suivi les étapes vues en cours et TP.

Au début nous avons travaillé sur l’ajout des employées et leurs affectations comme suite :

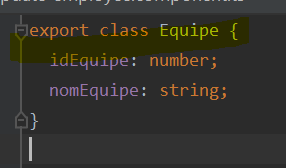
Nous avons ajouté le lien vers la page qui contient le formulaire d’affectation dans **app.component.html** comme suit :



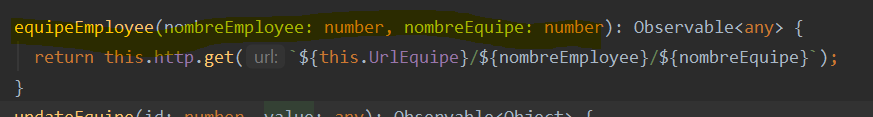
Par la suite nous avons ajouté l’objet Equipe dans **employee.ts :**



Aussi nous avons crée le fichier **equipe.ts** avec ses attributs comme suit :

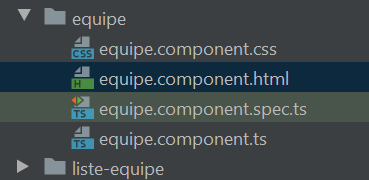


Dans le dossier service déjà crée dans le TP avec **ng g s service/employee** nous allons ajouter les fonctions dont nous avons besoin. Ici nous avons ajouté

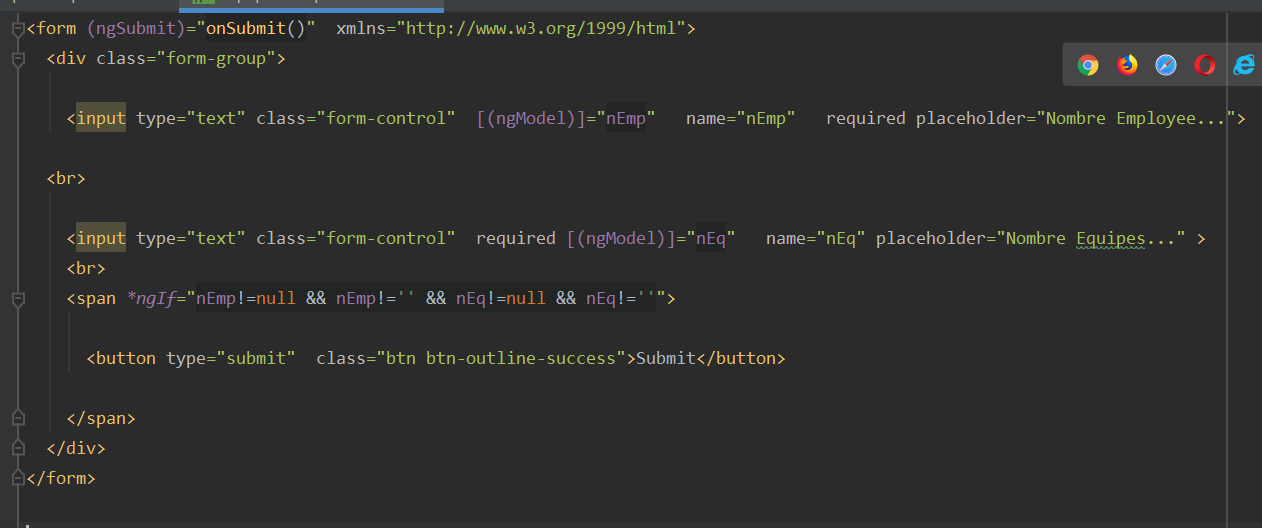


Par la suite nous avons ajouté le composant equipe comme suite :

**ng g c equipe** et nous avons le composant :

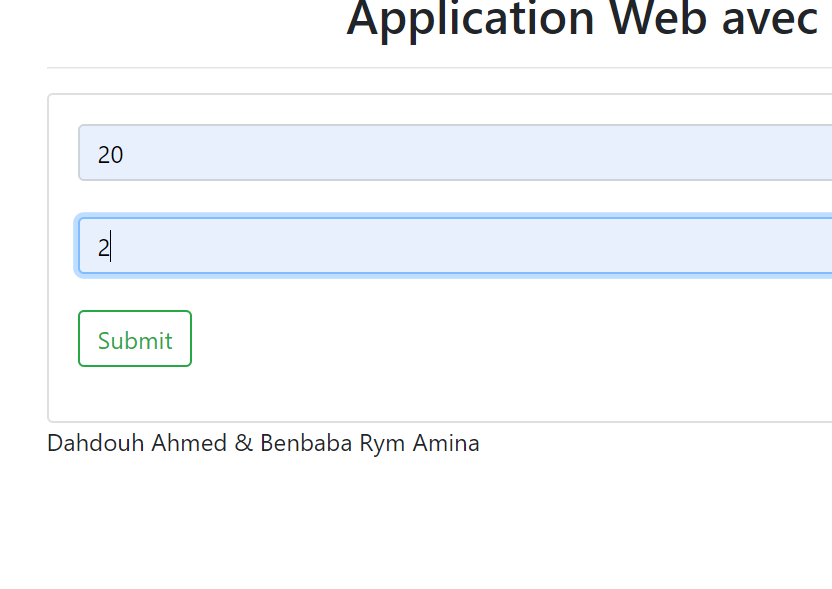


Dans le equipe.component.html nous avons le formulaire d’ajout :

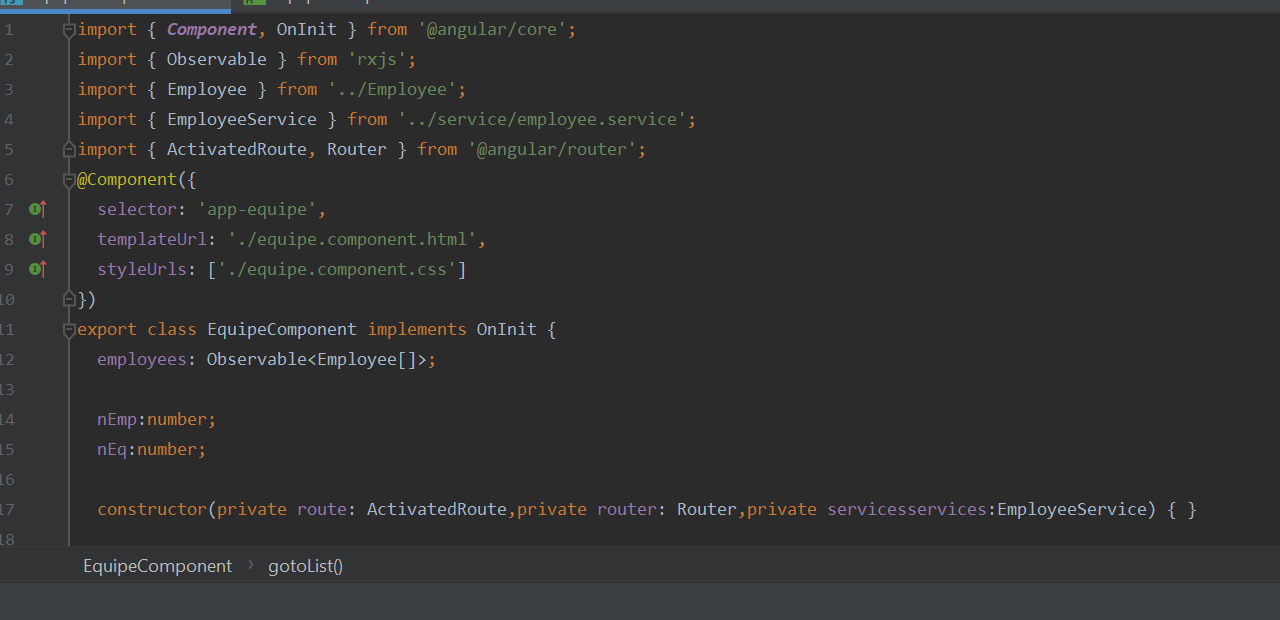
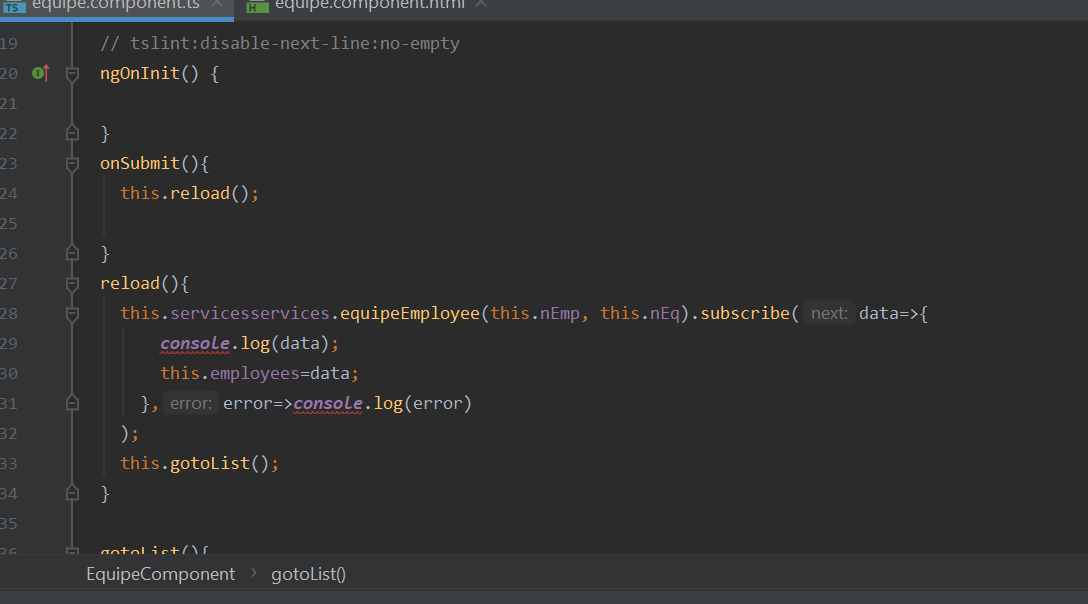


Il est à noté que notre formulaire n’affiche pas le button submit si et seulement si l’utilisateur à bien rempli les champs :





Dans equipe.component.ts nous avons :

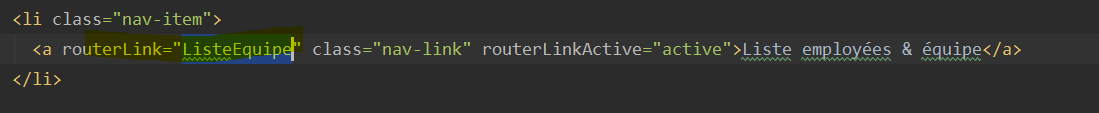
  

Dans **app-routing.moduls.ts** nous avons jouter le path





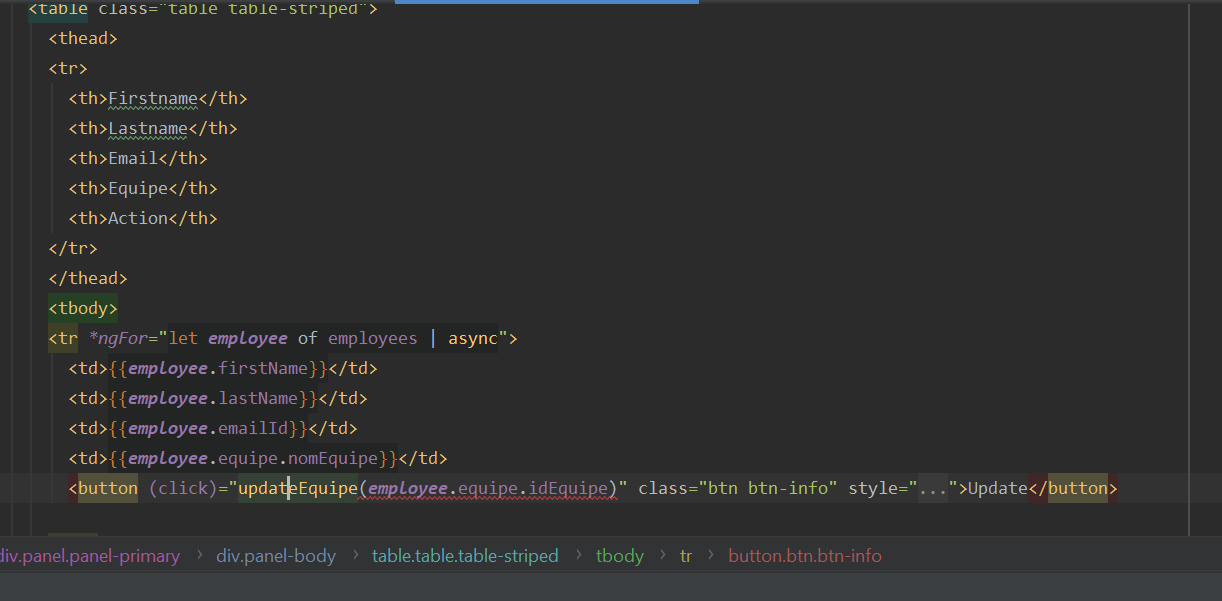
Une deuxième partie consiste la partie d’affichage des employées avec leurs équipes consiste à suivre les mêmes étapes précédentes alors dans un premier lieu nous avons ajouter le chemin dans **app.component.html**



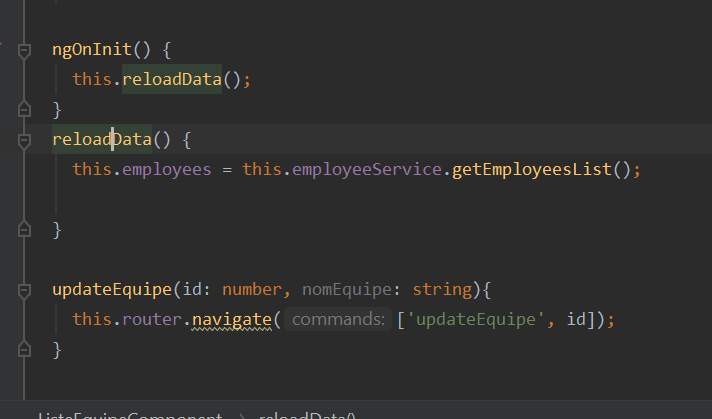
La même chose pour le path dans **app-routing.moduls.ts**

Un composant list-equipe à été crée aussi pour l’affichage donc nous avons e le html

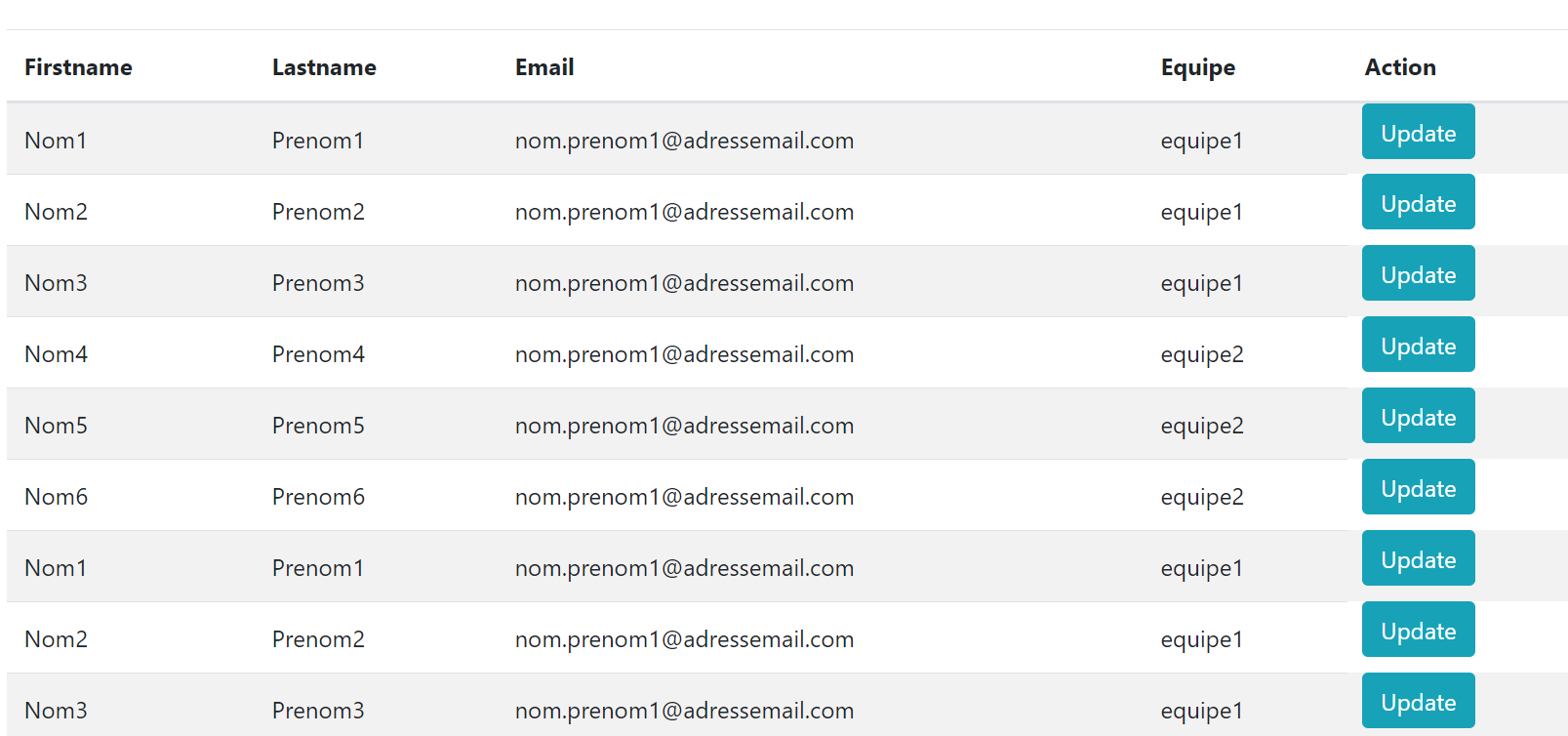


Et dans le ts



La fonction getEmployeesList est la même que la première pour l’affichage des employées.

Et donc nous avons eu les résultats suivants :

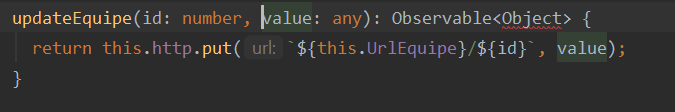


La dernière partie consiste les mises à jour des équipes des employées donc :

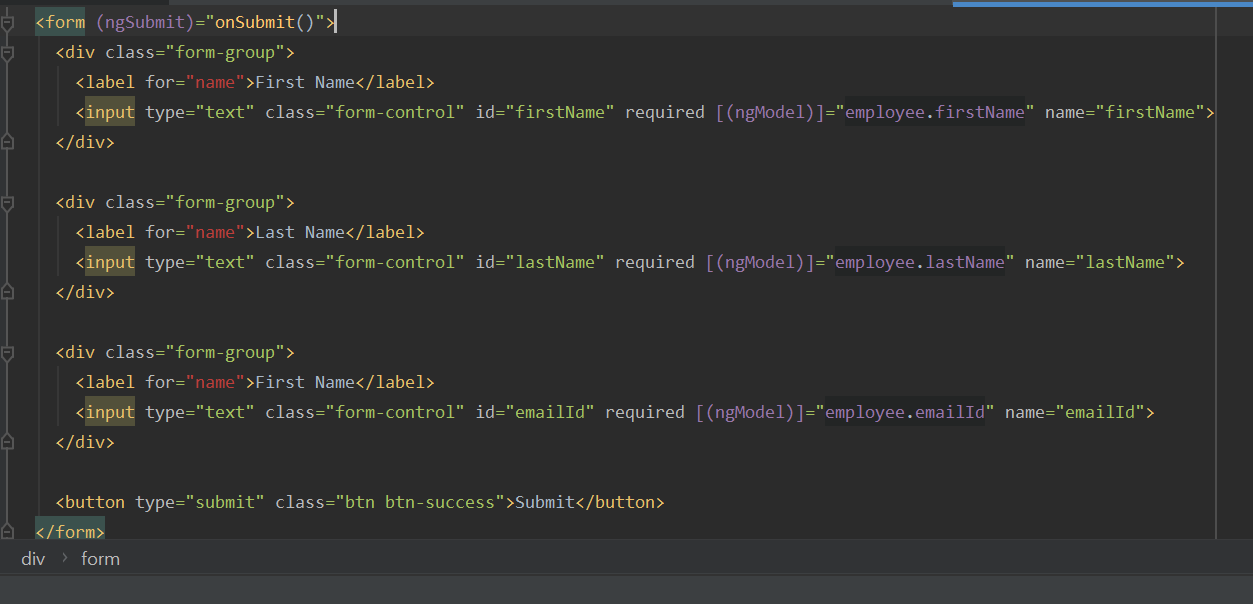
Dans le button UPDATE nous avons ajouter un lier comme suit :



Dans le service nous avons la fonction



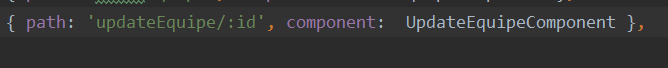
Le composant que nous avons crée update-employe contient dans l heml le formulaire :



Et dans le .ts la fonction de mise a jour :



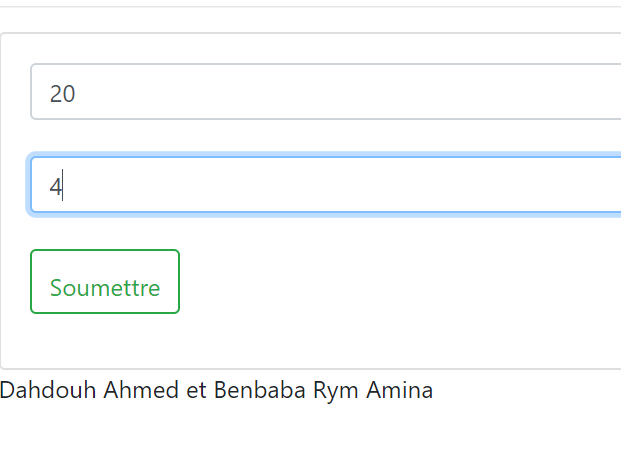
En principes c’est comme la fonction de update d’employée l’ancienne, donc pour les chemins nous avons ajouté

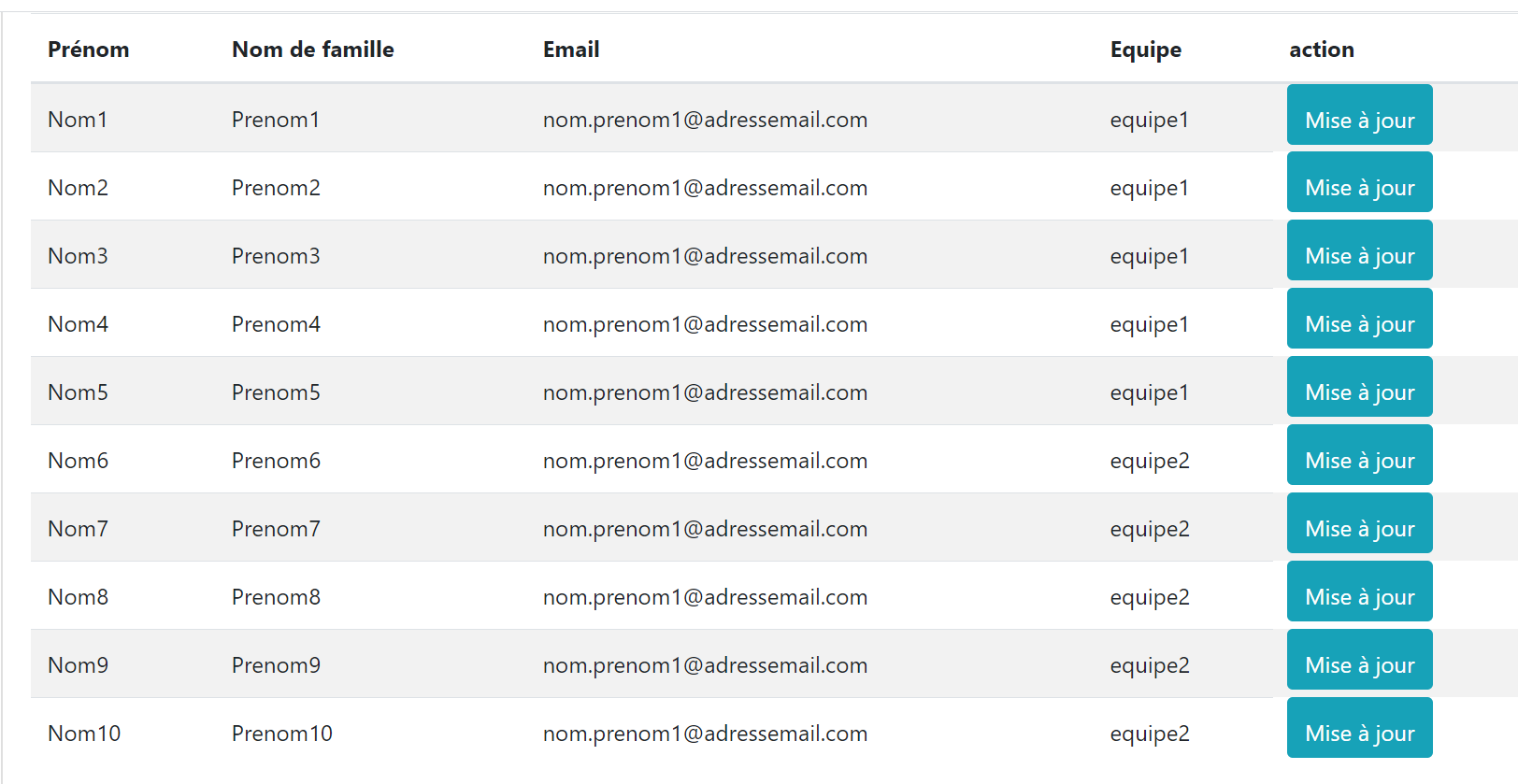
# **Interfaces**

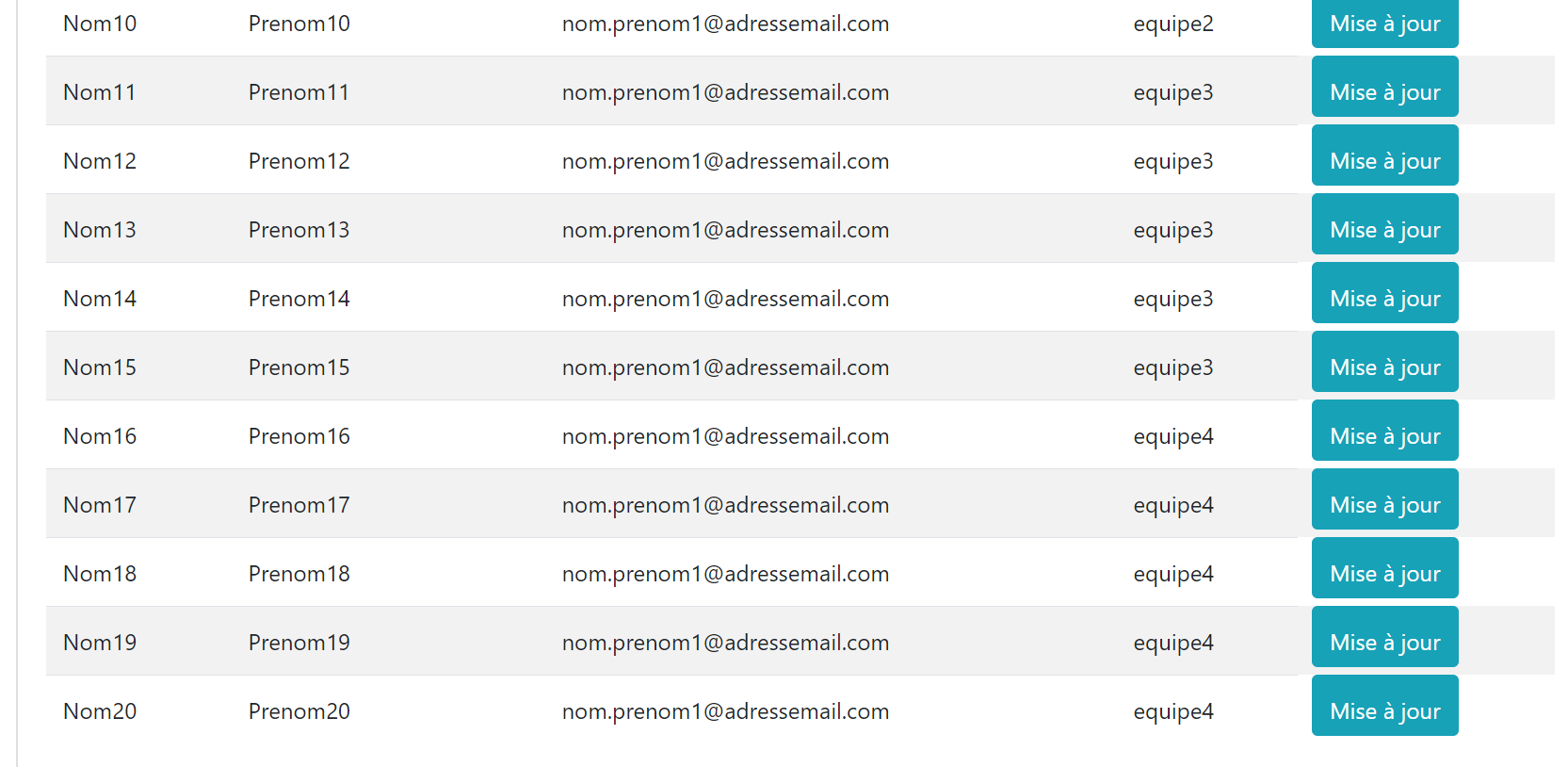
Pour les résultats que nous avons obtenus on a :

Nous allons définir 20 employées et 4 équipes :

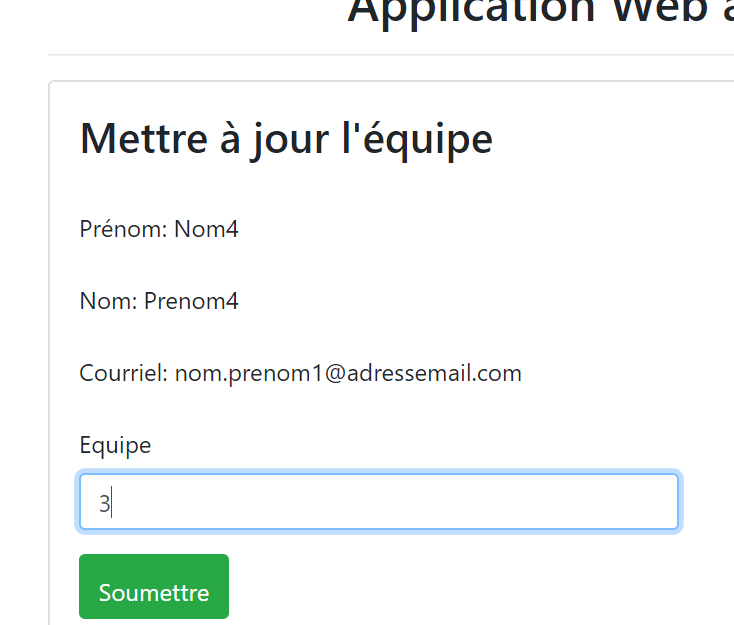


Apres un clique sur submit le composant qui contient l’affichage aura lieu :

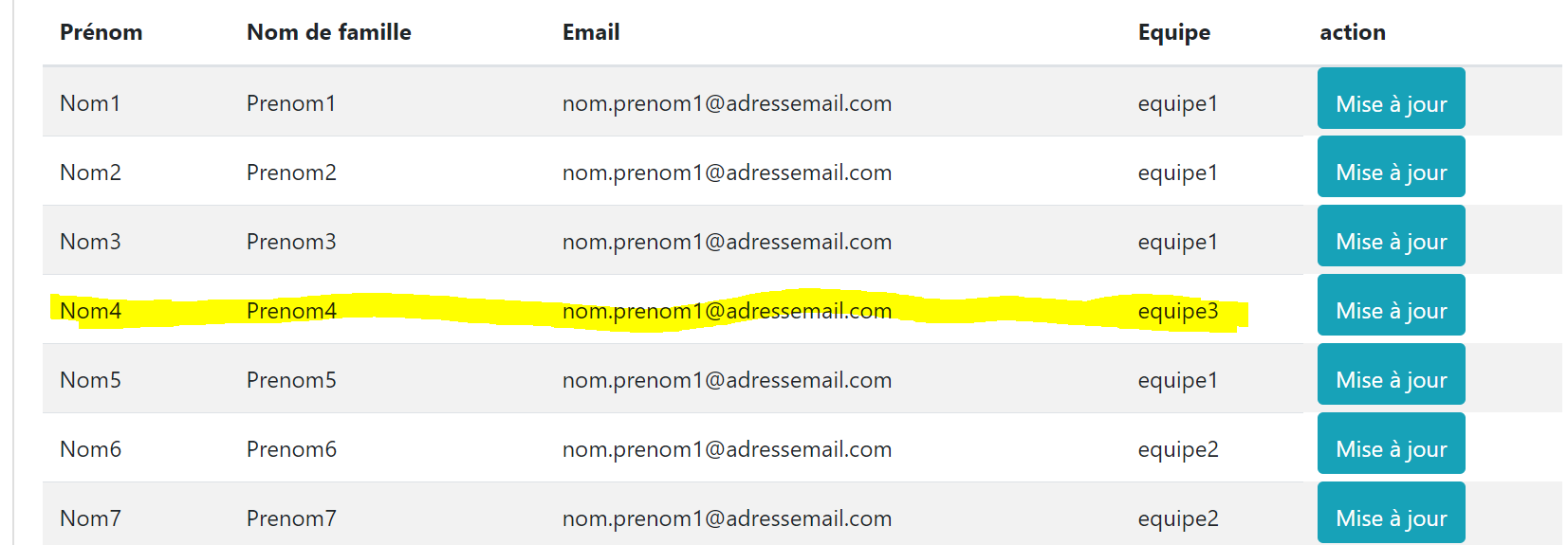




Pour une mise à jour nous allons chaoisi le (Nom 4) et changer l’équipe à 3



On obtient le résultat suivant :



Note : Lancer le serveur avec la commande **ng serve**