|  |  |
| --- | --- |
| Une image contenant texte, Police, logo, Graphique  Description générée automatiquement | Année Universitaire : 2023-2024  **Application côté client 1** |
| **Workshop :**  **Guard dans un projet Angular** | |

1. Installer le package suivant pour gérer l’authentification avec json web token (jwt)

**npm install -g json-server json-server-auth**

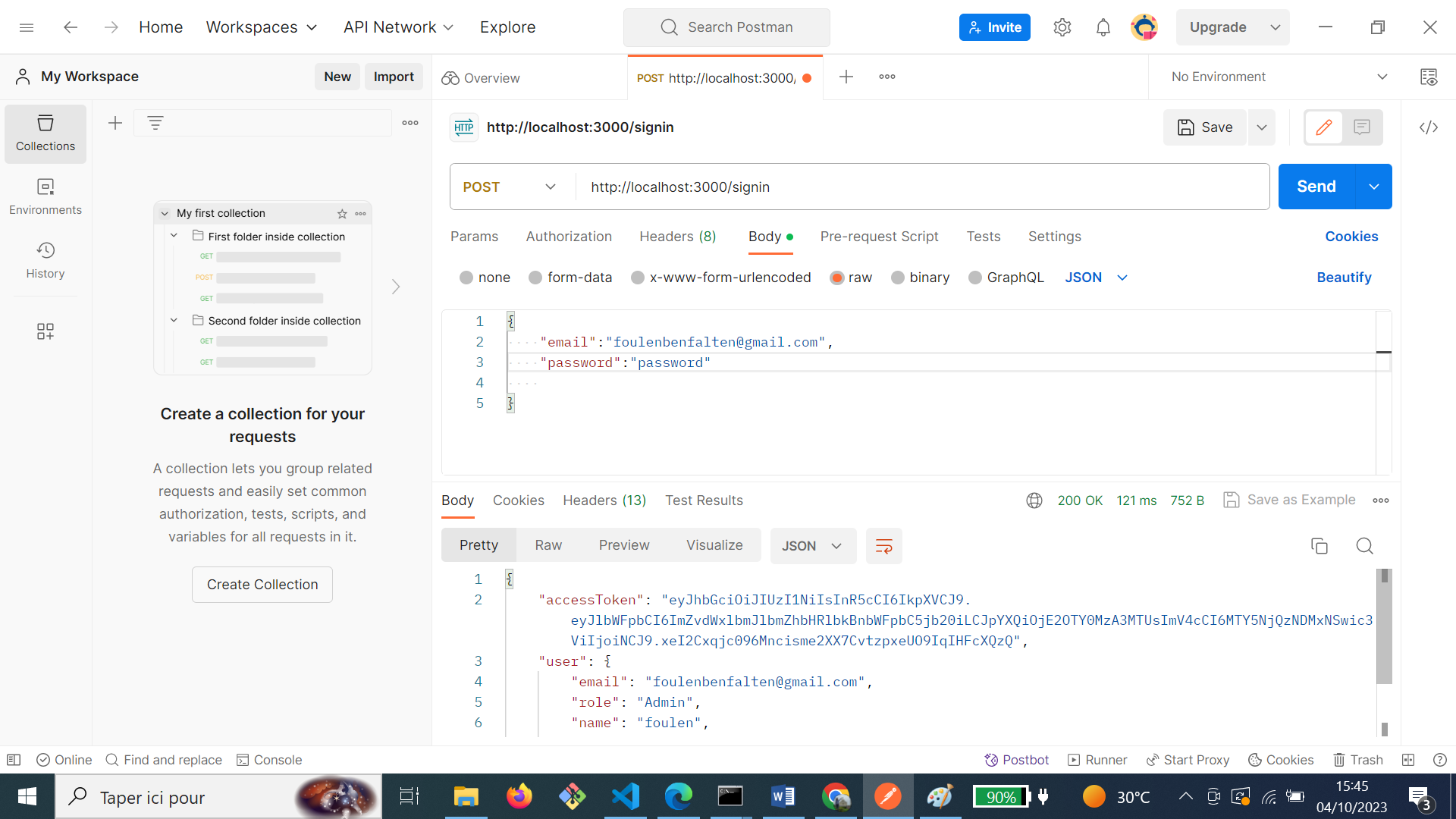
1. Créer un fichier db.json sous le projet
2. Lancer la commande **json-server-auth db.json**
3. Dans le fichier db.sjon créer une table **user** avec les attributs suivants email, pasword, role, name, surname
4. Ouvrez le logiciel Postmen, puis saisissez l'URL suivante dans la barre d'adresse. Ensuite, ajoutez les données suivantes dans le corps de la requête.

[**http://localhost:3000/register**](http://localhost:3000/register)

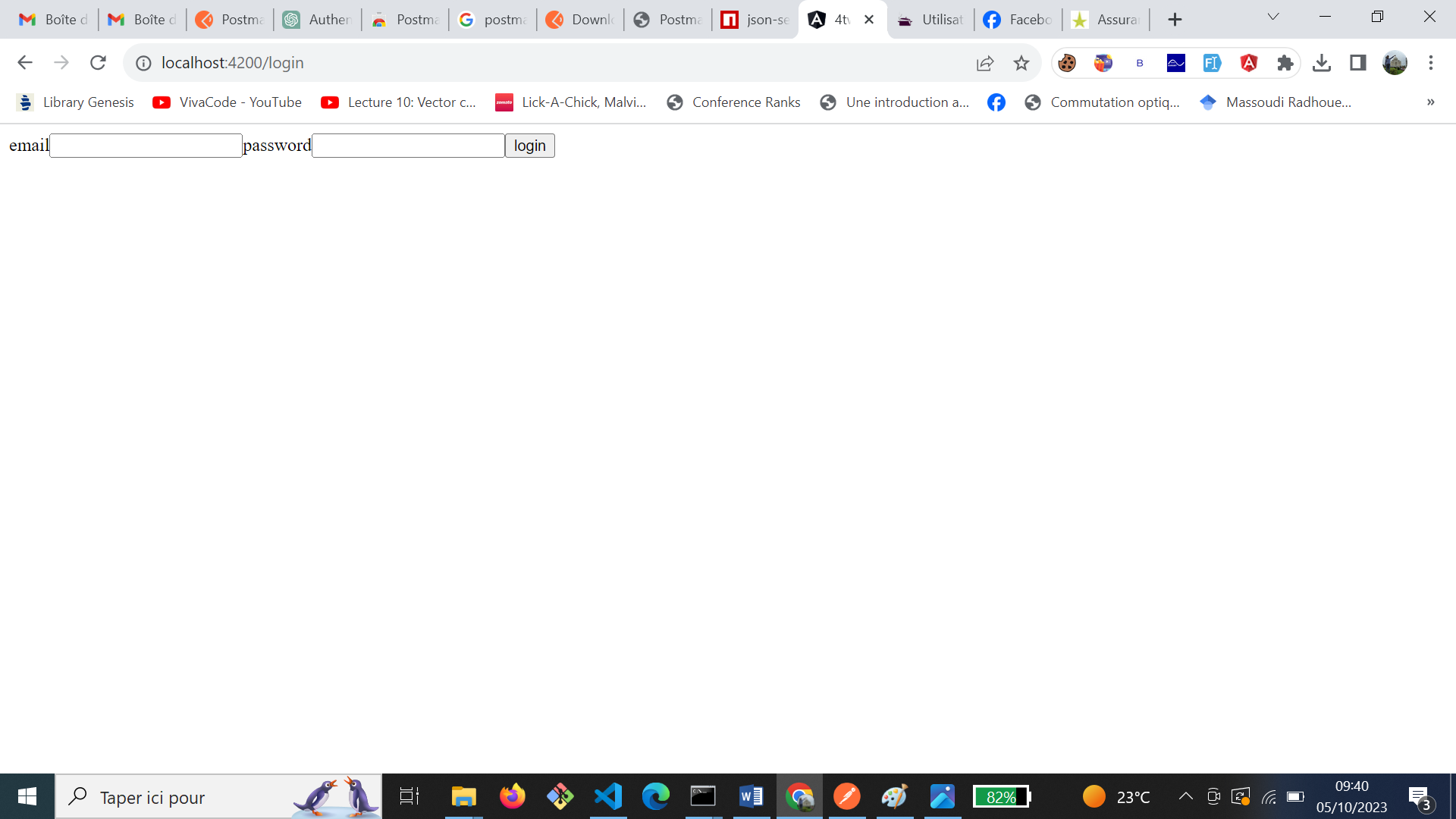
Une image contenant texte, Police, nombre, ligne

Description générée automatiquement

1. Pour s’authentifier il faut taper dans l’url **http://localhost:3000/signin**



1. Créer un composant login dedans on va afficher le formulaire comme indique la figure suivante :



1. Créer un service ‘**AuthentificationService’** et ajouter le code source suivant :

 constructor(private http:HttpClient) { }

  singin(data:any):Observable<User>{

    return this.http.post<User>('http://localhost:3000/signin',data);

  }

1. Ajouter le code source suivant dans le ts de composant login

constructor(private authService: AuthentificationService){

  }

  login(d:any): void {

    this.authService.singin(d).subscribe(

      (response) => {

        // Stockez le jeton JWT dans le stockage local

       localStorage.setItem('access\_token', response.accessToken);

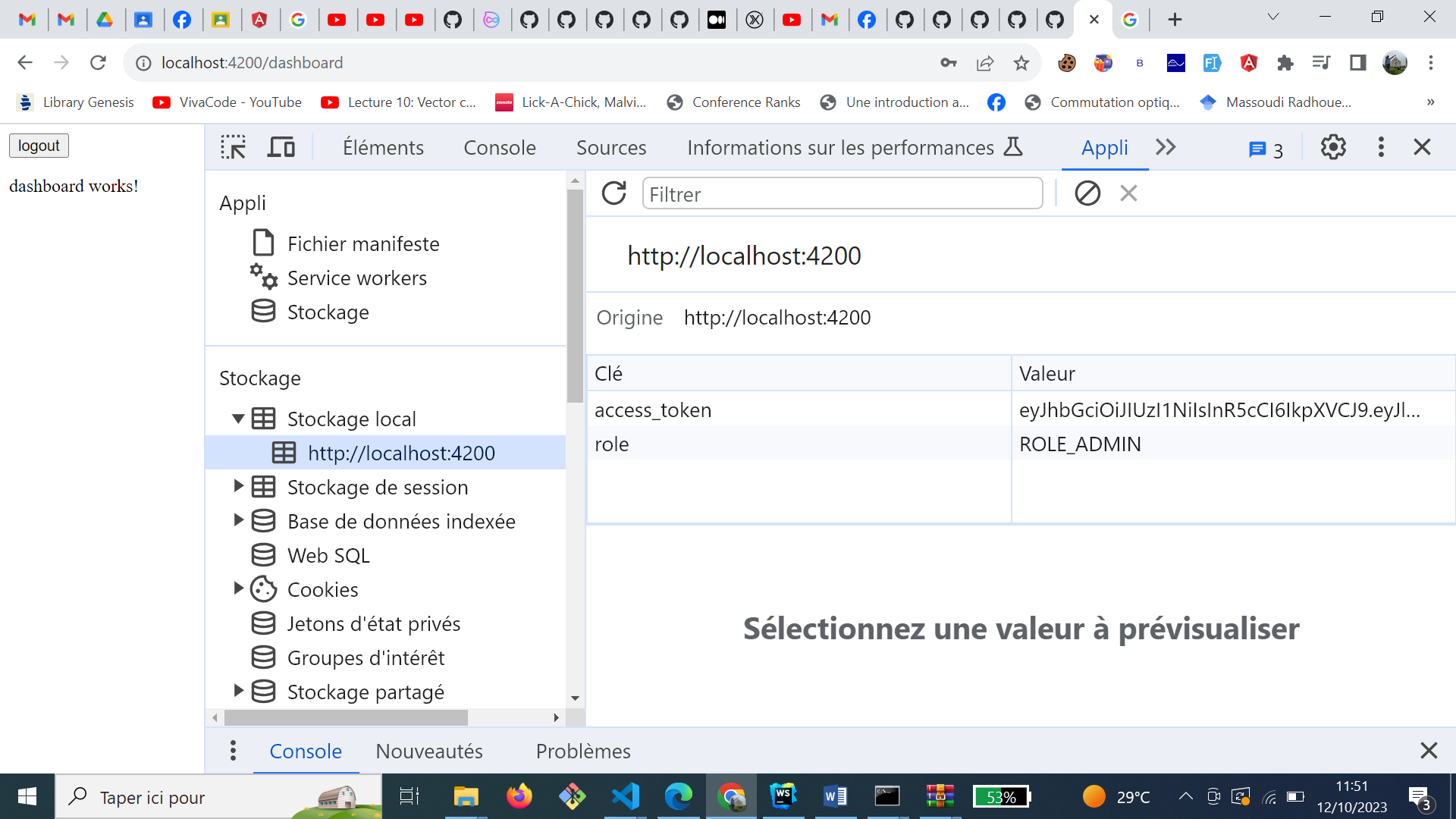
localStorage.setItem('role', response.role);

      }

    );

  }

1. Appelez la méthode login dans le hml du composant login
2. Enter le login et password dans l’interface puis un tekon sera stocké dans le local Storage de navigateur comme indique la figure suivante



10. implémenter la méthode **\_is\_logged()** qui permet de vérifier si l’utilisateur connecté ou pas dans le service **AuthentificationService**

\_is\_logged(): boolean {

  return !!localStorage.getItem('access\_token');

  }

10. on va ajouter l’obligation d’authentification pour cela lancer la commande suivante pour générer un **guard ‘ng generate guard authentification’** et choisir la méthode **canactivate**

11. appeler la méthode **\_is\_logged()** dans la méthode **canActivate()** dans le fichier **authentification.guard.ts**

constructor(private authService: AuthentificationService) {

  }

  canActivate(

    route: ActivatedRouteSnapshot,

    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {

    if (this.authService.\_is\_logged() ) {

      return  true

         }

    return false;

  }

1. Ajouter le **canActivate** aux routes nécessitantes une authentification comme montre la figure suivante.

{    path: 'dashboard', component: DashboardComponent, canActivate: [AuthentificationGuard]}

1. Nous allons attribuer un rôle pour accéder à la page "**dashboard**" et mettre en place la méthode "**getRole(role: String)"** dans le service **AuthentificationService** pour vérifier le rôle de l'utilisateur.

getRole(role:String) {

    this.roleAs = localStorage.getItem('role');

        if(this.roleAs==role)

      return true;

    return false

  }

1. Créez un garde en utilisant la commande **'ng generate guard roleGuard**' et sélectionnez la méthode 'canActivate'.
2. Ajouter le code source suivant dans le fichier **role.guard.ts**

 constructor(private authService:AuthentificationService ) {

  }

  canActivate(

    route: ActivatedRouteSnapshot,

    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {

    if (this.authService.getRole('ROLE\_ADMIN') ) {

      return true; // L'utilisateur est authentifié, autorisez l'accès à la route (rôle admin)

    }

    this.authService.logOut();

    return false;

  }

1. Ajouter le **guard** rôle dans le fichier de routing

 {

    path: 'dashboard', component: DashboardComponent, canActivate: [AuthentificationGuard,RoleGuard]

  },

1. Testez le code