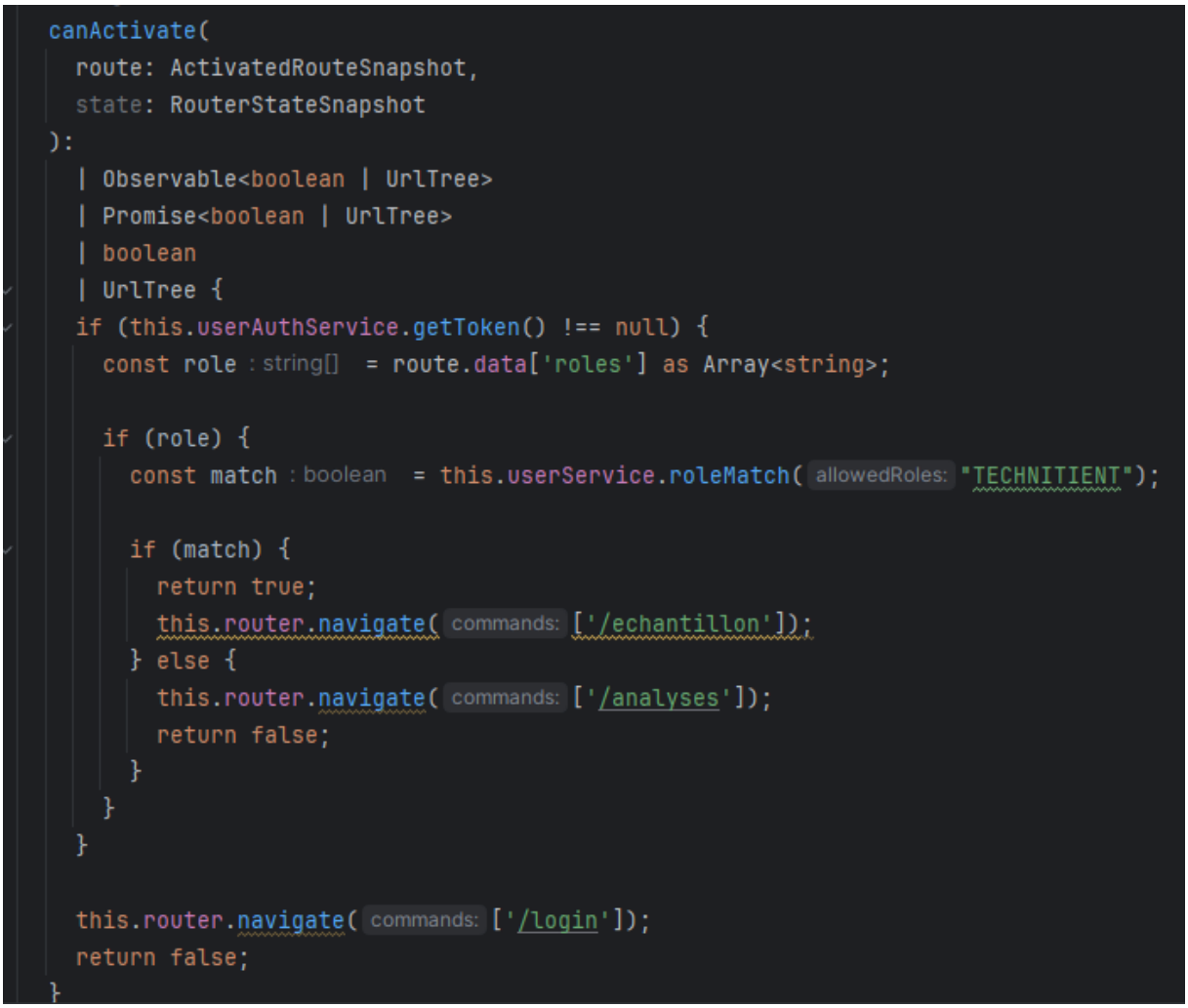

Le `AuthGuard` est un garde de route (route guard) dans application. Il implémente l'interface `CanActivate`, ce qui signifie qu'il peut décider si une route peut être activée ou non en fonction de certaines conditions.

Dans ce cas, le `AuthGuard` vérifie si un utilisateur est authentifié en vérifiant s'il existe un jeton d'authentification. S'il y a un jeton, il vérifie ensuite si l'utilisateur possède le rôle requis pour accéder à la route. Si l'utilisateur possède le rôle requis, la route est activée et l'utilisateur est redirigé vers la page appropriée. Sinon, l'utilisateur est redirigé vers une autre page en fonction de son rôle ou vers la page de connexion s'il n'est pas authentifié.

L'intercepteur AuthInterceptor est un intercepteur HTTP dans une application Angular. Il implémente l'interface HttpInterceptor, ce qui lui permet d'intercepter les requêtes HTTP sortantes et les réponses HTTP entrantes.

Dans cet intercepteur, plusieurs tâches sont effectuées :

Vérification de l'authentification : L'intercepteur vérifie si la requête HTTP a l'en-tête No-Auth défini sur True. Si c'est le cas, il passe la requête telle quelle sans ajout de jeton d'authentification.

Ajout du jeton d'authentification : Si la requête n'a pas l'en-tête No-Auth défini sur True, l'intercepteur récupère le jeton d'authentification à partir du service UserAuthService et l'ajoute à l'en-tête Authorization de la requête.

Gestion des erreurs HTTP : L'intercepteur utilise catchError pour intercepter les erreurs HTTP. S'il reçoit une erreur HTTP 401 (Non autorisé), il redirige l'utilisateur vers la page de connexion. S'il reçoit une erreur HTTP 403 (Interdit), il redirige l'utilisateur vers une autre page (dans ce cas, /analyses). Dans tous les autres cas d'erreur, il renvoie simplement un message d'erreur générique.

```
public roleMatch(allowedRoles): boolean {
  let isMatch : boolean  = false;
  const userRoles: any = this.userAuthService.getRoles();

  if (userRoles != null && userRoles) {
    for (let i : number  = 0; i < userRoles.length; i++) {
      for (let j : number  = 0; j < allowedRoles.length; j++) {
        if (userRoles[i].roleName === allowedRoles[j]) {
          isMatch = true;
          return isMatch;
        } else {
          return isMatch;
        }
      }
    }
  }
}
```

La méthode **roleMatch** est une fonctionnalité d'un service Angular qui vérifie si les rôles de l'utilisateur actuellement authentifié correspondent à une liste de rôles autorisés. Elle prend en entrée un tableau de rôles autorisés et compare chaque rôle autorisé avec les rôles de l'utilisateur. Cependant, son implémentation actuelle comporte des erreurs de logique, notamment dans les boucles **for**. La méthode ne parcourt pas correctement tous les rôles de l'utilisateur ni tous les rôles autorisés, et elle retourne **false** dès qu'aucun match n'est trouvé pour le premier rôle de l'utilisateur comparé avec le premier rôle autorisé. Il est donc nécessaire de réviser la logique de la méthode pour garantir un comportement correct et complet de la vérification des rôles.