Formation Angular

TP N°13: L'internationalisation (i18n)

SOMMAIRE

1. Pré-requis :
   - Réaliser les TPs 1-12 en premier.

2. Objectifs
   ✓ Implémenter le principe de l'internationalisation (i18n) avec la librairie **ngx-translate.**

3. Développement de la couche front

a. Importer le projet TP13

- Créer un nouveau dossier (TP13) et copier dans ce dernier le contenu du dossier TP12.
- Lancer les commandes suivantes :
  - **cd tp13**
  - **npm install**

b. Installation de la librairie ngx-translate

- Lancer la commande **npm i @ngx-translate/core –save**
- Lancer la commande **npm i @ngx-translate/http-loader –save**

c. Importation des modules TranslateLoader, TranslateModule et TranslateHttpLoader

- Modifier le fichier app.module.ts comme suit :

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { HttpClient, HttpClientModule } from '@angular/common/http';
import { AuthComponent } from './auth/auth.component';
import { NavbarComponent } from './navbar/navbar.component';
import { authInterceptorProviders } from './interceptors/auth.interceptor';
import { WelcomeComponent } from './welcome/welcome.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import {MatPaginatorModule} from '@angular/material/paginator';
import {MatProgressSpinnerModule} from '@angular/material/progress-spinner';
import {MatCardModule} from '@angular/material/card';
import {MatTableModule} from '@angular/material/table';
import {MatButtonModule} from '@angular/material/button';
import {MatSnackBarModule} from '@angular/material/snack-bar';
import { Product1Component } from './product1/product1.component';
import { Product2Component } from './product2/product2.component';
import { ProductEditComponent } from './product-edit/product-edit.component';

import { TranslateLoader, TranslateModule } from '@ngx-translate/core';
```

```
import { TranslateHttpLoader } from '@ngx-translate/http-loader';

const materialComponents=[
  MatPaginatorModule,
  MatProgressSpinnerModule,
  MatCardModule,
  MatTableModule,
  MatButtonModule,
  MatSnackBarModule
]


@NgModule({
  declarations: [
    AppComponent,
    AuthComponent,
    NavbarComponent,
    WelcomeComponent,
    EmpListComponent,
    EmpCreateComponent,
    Product1Component,
    Product2Component,
    ProductEditComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    HttpClientModule,
    TranslateModule.forRoot({
      loader: {
        provide: TranslateLoader,
        useFactory: httpTranslateLoader,
        deps: [HttpClient]
      }
    }),
    ReactiveFormsModule,
    BrowserAnimationsModule,
    materialComponents
  ],
  providers: [authInterceptorProviders],
  bootstrap: [AppComponent]
})
export class AppModule { }
export function httpTranslateLoader(http: HttpClient) {
  return new TranslateHttpLoader(http);
}
```

d. Modifier le composant navbar

- Modifier le fichier navbar.component.ts comme suit :

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { TranslateService } from '@ngx-translate/core';
import { TokenStorageService } from '../services/token-storage.service';

@Component({
  selector: 'app-navbar',
  templateUrl: './navbar.component.html',
  styleUrls: ['./navbar.component.css']
})
export class NavbarComponent implements OnInit {

  isLoggedIn = false;
  isAdmin = false;
  isClient = false;
  username?: string;

  constructor(public translate:TranslateService,private tokenStorageService:
TokenStorageService, private route: ActivatedRoute, private router: Router) {
    translate.addLangs(['en', 'fr']);
    translate.setDefaultLang('en');

  }

  ngOnInit(): void {

    console.log("tokrn",this.tokenStorageService.getTokenValue());
     if (this.tokenStorageService.getTokenValue() != null)
     this.isLoggedIn=true;

    if (this.isLoggedIn) {
      const user = this.tokenStorageService.getUsername();
      this.isAdmin = <boolean>this.tokenStorageService.hasRole('ADMIN');
      this.isClient = <boolean>this.tokenStorageService.hasRole('CLIENT');
      this.username = <string>this.tokenStorageService.getUsername();
    }
  }
  logout(): void {
    this.tokenStorageService.signOut();
    this.router.navigate([{ outlets: { primary: 'login', contenu: null } }]);
  }


  switchLang(lang: string) {
```

```
    this.translate.use(lang);
  }

}
```

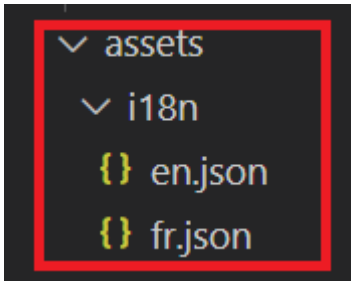
- Modifier la page navbar.component.html comme suit :

```
<nav class="navbar navbar-expand navbar-dark bg-dark">
    <a href="#" class="navbar-brand">CIGMA</a>
    <ul class="navbar-nav mr-auto" routerLinkActive="active">
        <li class="nav-item" *ngIf="isAdmin || isClient">
            <a class="nav-link" [routerLink]="['/',{ outlets: { primary:
['navbar'],contenu: ['employees'] } }]">{{'employees.navbar.menu' | translate}}</a>
        </li>

        <li class="nav-item" *ngIf="isAdmin">
            <a class="nav-link" *ngIf="isLoggedIn" routerLink="user">Gestion des
utilisateurs</a>
        </li>

        <li class="nav-item" >
            <a class="nav-link" [routerLink]="['/',{ outlets: { primary:
['navbar'],contenu: ['product1'] } }]">Gestion des produits</a>
        </li>

        <li class="nav-item" >
            <a class="nav-link" [routerLink]="['/',{ outlets: { primary:
['navbar'],contenu: ['product2'] } }]">Gestion des produits avec Resolver</a>
        </li>
    </ul>

    <ul class="navbar-nav ml-auto" *ngIf="isLoggedIn">
        <li class="nav-item">
            <a class="nav-link" routerLink="profile">{{ username }}</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" (click)="logout()">LogOut</a>
        </li>

          <li class="nav-item">
            <select class="form-control" #selectedLang
(change)="switchLang(selectedLang.value)">
              <option *ngFor="let language of translate.getLangs()"
[value]="language"
                [selected]="language === translate.currentLang">
                {{ language }}
```

```
            </option>
          </select>
        </li>
    </ul>
</nav>
```

e. Création des fichiers language.json

- Dans le dossier assets, créer le dossier i18n et créer les deux fichiers en.json et fr.json comme le montre l'imprime écran suivant :



- en.json :

```
{
    "login": "login",
    "password": "password",
    "employees.navbar.menu":"Employees management"
}
```

- fr.json :

```
{
    "login": "Identifiant",
    "password": "Mot de passe",
    "employees.navbar.menu":"Gestion des employés"
}
```

f. Remplacement des textes statiques aux niveau des pages par leurs clés

- Au niveau de vos pages html, remplacer les textes statiques par leur clés en utilisant l'expression suivante : **{{'votre_clé|translate'}}.** Par exemple, dans la page html de navbar remplacer le menu « Gestion des employés » par :

```
{{'employees.navbar.menu' | translate}}
```

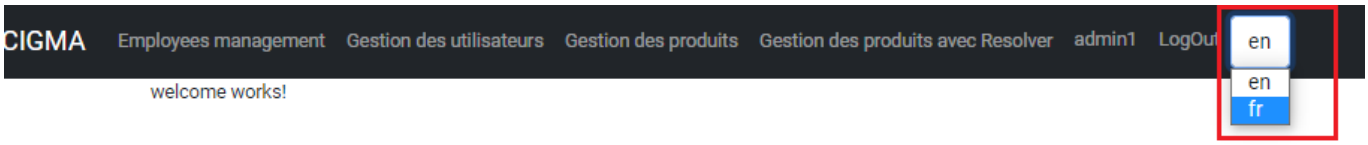
Et dans la page de login, remplacer username et password par :

```
 placeholder="{{'login' | translate}}"
```

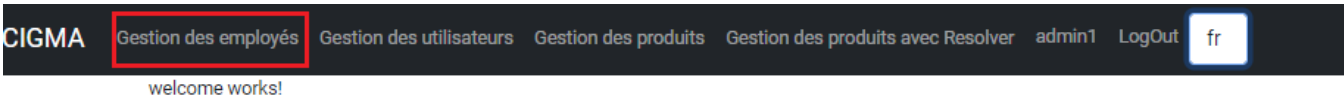
```
placeholder="{{'password' | translate}}"
```

4. Tests

- Lancer l'application : **ng serve** et accéder au lien http://localhost:4200.
- Au niveau de la page d'authentification, entrer le compte « admin1/admin1 » et cliquer sur LOGIN. La page suivante s'affiche :



- Dans la liste déroulante ci-dessus, changer la langue et vérifier que le texte correspondant au menu « Employees management » change selon la langue choisie :



- De même au niveau de la page d'authentification. Cliquer sur « LogOut » et vérifier que les noms des champs au niveau du formulaire ont été également mis à jour :

- Maintenant, vous pouvez codifier tous vos messages statiques au niveau de vos pages HTML de la même façon.

Fin du TP 13.