Exemple d'utilisation du ReactiveFormsBuilder

Dans connexion.module.ts :

Remplacer dans l'import FormsModule par ReactiveFormsModule
(import {ReactiveFormsModule} from '@angular/forms';)

Dans connexion.page.ts :

1. Déclarer le module dans le constructeur : public formBuilder: FormBuilder
2. Importer les modules nécessaires :
import {FormBuilder, FormControl, FormGroup, Validators} from '@angular/forms';
3. Déclarer la variable connexionForm (de type FormGroup) comme attribut de classe :
connexionForm: FormGroup;
4. Construire le formulaire dans le constructeur :

```
this.connexionForm = this.formBuilder.group({
                email: new FormControl('', Validators.compose([
                 Validators.required
        ])),
                 mdp: new FormControl('', Validators.compose([
                  Validators.required
        ]))
});
```

5. Créer les messages d'erreur à afficher si les contraintes ne sont pas respectées :

en dessous de l'attribut de classe moteurRechercheForm: FormGroup :

```
error_messages = {

        email: [
                {type: 'required', message: 'L\'adresse email est requise.'}
        ],
        mdp: [
                {type: 'required', message: 'Le mot de passe est requis.'}
        ]
}
```

```
connexionForm: FormGroup;
error_messages = {
    email: [
        {type: 'required', message: 'L\'adresse email est requise.'},
        {type: 'pattern', message: 'L\'adresse email saisie n\'est pas valide.'}
    ],
    mdp: [
        {type: 'required', message: 'Le mot de passe est requis.'}
    ]
}
constructor(public formBuilder: FormBuilder) {
    this.connexionForm = this.formBuilder.group({
        email: new FormControl('', Validators.compose([
            Validators.required,
            Validators.pattern(/^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".
+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))
$/)
        ])),
        mdp: new FormControl('', Validators.compose([
            Validators.required
        ]))
    });
}
```

Dans connexion.page.html :

Créer un formulaire avec la balise form et renseigner le nom du formulaire utilisé dans l'attribut formGroup :

<form [formGroup]= "connexionForm">

</form>

Ensuite, il faut créer les champs qu'on a précédemment ajoutés au connexionForm (dans le

constructeur) et, en dessous de chaque champs, ajouter la div des messages d'erreur suivante :

<ng-container *ngFor="let error of error_messages.email">

<div class="error-message" *ngIf="connexionForm.get('email').hasError(error.type) && (connexionForm.get('email').dirty || connexionForm.get('email').touched)">
{{error.message}}

</ng-container>

<ng-container *ngFor="let error of error_messages.mdp">

<div class="error-message" *ngIf="connexionForm.get('mdp').hasError(error.type) && (connexionForm.get('mdp').dirty || connexionForm.get('mdp').touched)">
{{error.message}}

</ng-container>

Et enfin, ajouter un bouton de type submit pour soumettre le formulaire :

<ion-button [disabled]="connexionForm.valid" shape="round" expand="full"

(click)="frechercheAnnonces()">Rechercher</ion-button>

```
<form [formGroup]="connexionForm">
  <ion-item>
    <ion-label position="floating">Adresse email</ion-label>
    <ion-input type="text" formControlName="email"></ion-input>
  </ion-item>
  <div class="error-messages">
    <ng-container *ngFor="let error of error_messages.email">
      <div class="error-message" *ngIf="connexionForm.get('email').hasError(error.type) &&
(connexionForm .get('email').dirty || connexionForm.get('email').touched)">
        {{error.message}}
      </div>
    </ng-container>
  </div>
  <ion-item>
    <ion-label position="floating">Mot de passe</ion-label>
    <ion-input type="password" formControlName="mdp"></ion-input>
```

```
  </ion-item>
  <div class="error-messages">
    <ng-container *ngFor="let error of error_messages.mdp">
      <div class="error-message" *ngIf="connexionForm .get('mdp').hasError(error.type) &&
(connexionForm .get('mdp').dirty || connexionForm.get('mdp').touched)">
        {{error.message}}
      </div>
    </ng-container>
  </div>
  <ion-button [disabled]="!connexionForm .valid" shape="round" color="primary" expand="full"
(click)="connexion()">Valider</ion-button>
</form>
```

Après avoir fait tout cela, il ne restera plus qu'à écrire la fonction connexion() qui va récupérer les identifiants saisis dans le formulaire et effectuer le traitement nécessaire.

```
public connexion() {
    var email = this.connexionForm.value.email;
    var mdp = this.connexionForm.value.mdp;

    // Suite du code ici...

}
```