# Angular

Tahaluf  Training Center  2021

شركــة تحالــف الإمــارات للحـــول التقنيــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

# Chapter 4

# Spinner

Add the spinner in login page.

First , you must Import the library in **auth.module.ts** :

```
import {MatProgressSpinnerModule} from  '@angular/ma
terial/progress-spinner';
```

Define the MatProgressSpinnerModule in import.

**In login.component.html:**

```
<mat-progress-spinner class="example-margin"
[color] = "color"[mode] = "mode"[value] = "value" >
  </mat - progress - spinner >
```

**In login.component.ts:**

Because the color, mode and value are a variables you must define it in ts:

```
color: ThemePalette = 'primary';
mode: ProgressSpinnerMode = 'indeterminate';
value = 50;
```

To use ThemePalette , ProgressSpinnerMode  import in typescript file:

```typescript
import { ThemePalette } from '@angular/material/core';
import { ProgressSpinnerMode } from '@angular/material/progress-spinner';
```

If the user enter the valid username and password show the loader and if not , show Login:

```html
<button [disabled]="!emailFormControl.valid
||!passwordFormControl.valid" mat-raised-button
(click)="submit()"style="flex:1;width: 100% ;margin-top:
50px;" color="primary">
<ng-container *ngIf="ShowLoader">
<mat-progress-spinner class="example-margin"
[color]="color" [mode]="mode" [value]="value">
</mat-progress-spinner></ng-container>
<ng-container *ngIf="!ShowLoader">Login
</ng-container></button>
```

# Spinner

**In login.component.ts:**

```
submit(){
    //Go to Loader
    this.ShowLoader = true;
    setTimeout(() => {
        this.ShowLoader = false;
    }
    ), 2000
}
```

Another way to add the spinner .. using this library: ng-x spinner

first you must install this package:
**npm i ng-x spinner**

Then Import NgxSpinnerModule in in the root module(AppModule):
```
import { NgxSpinnerModule } from "ngx-spinner";
```

And in import section add NgxSpinnerModule.


Then:  add NgxSpinnerService service wherever you want to use the ngx-spinner (**In login.components.ts**)

```
import { NgxSpinnerService } from "ngx-spinner";

private spinner: NgxSpinnerService
```

If you want to show the spinner use:

```
this.spinner.show()
and to hide the spinner:
this.spinner.hide();
```

Add in app.component.html

```
<ngx-spinner bdColor="rgba(0, 0, 0, 0.8)"
size="medium" color="#fff" type="square-jelly-
box" [fullScreen] = "true" > <p
style="color: white" > Loading... </p></ngx -
spinner >
```

# Chapter 4

To go from login page to register page you must use routing.

Add in login.component.html:

```html
</div >
<p (click)="goToRegisterPage()" style="cursor: pointer;
text-align: center;font-size: 1.7em;margin-top: 40px; ">
Create new account</p>
    </div >
```

# How to navigate from pages

**In login.component.ts**

In constructer parameter:

```
private router:Router
```

Add this function:

```
goToRegisterPage(){
    this.router.navigate(['register'])
}
```

# Chapter 4

We will use form group.

Define an instance of form group.

**In register.component.ts** and inside this instance will declared instance of form group

# Create and validate form using form group

**In register.component.ts:**

```typescript
export class RegisterComponent implements OnInit {

registerForm: FormGroup = new FormGroup({
fullName: new FormControl('', [Validators.required]),
email: new FormControl('', [Validators.required,
Validators.email]),
address: new FormControl('', [Validators.required]),
phoneNumber: new FormControl(''),
password: new FormControl('', [Validators.required,
Validators.minLength(8)])
    })
```

# Create and validate form using form group

```html
<form [formGroup]="registerForm" (submit)="submit()">
    <mat-form-field class="example-full-width">
        <mat-label>FullName</mat-label>
        <input matInput formControlName="fullName" style=
"font-size: x-large;" placeholder="Full Name ">
        <mat-
error *ngIf="registerForm.controls.fullName.hasError('req
uired')">
            fullName is <strong>required</strong>
        </mat-error>
    </mat-form-field>
```

# Create and validate form using form group

```html
<mat-form-field class="example-full-width">
    <mat-label>Email</mat-label>
    <input matInput formControlName="email" style="font-size: x-large;" placeholder="Ex. pat@example.com">
    <mat-error
        *ngIf="registerForm.controls.email.hasError('email') && !registerForm.controls.email.hasError('required')">
        Please enter a valid email address
    </mat-error>
    <mat-error *ngIf="registerForm.controls.email.hasError('required')">
        Email is <strong>required</strong>
    </mat-error>
</mat-form-field>
```

```html
<mat-form-field class="example-full-width">
    <mat-label>Address</mat-label>
    <input matInput formControlName="address" style="font-size: x-large;" placeholder="Ex. WI Street">
    <mat-error *ngIf="registerForm.controls.address.hasError('required')">
        address is <strong>required</strong>
    </mat-error>
</mat-form-field>
```

# Create and validate form using form group

```
<mat-form-field class="example-full-width">
    <mat-label>Phone Number</mat-label>
    <input matInput formControlName="phoneNumber" st
yle="font-size: x-
large;" placeholder="Ex. 077 988 9870">
</mat-form-field>
```

# Create and validate form using form group

```html
<mat-form-field class="example-full-width">
    <mat-label>Password</mat-label>
    <input type="password" matInput formControlName=
"password" style="font-size: x-large;">
    <mat-error
        *ngIf="registerForm.controls.password.hasErr
or('minlength') && !registerForm.controls.password.hasEr
ror('required')">
            Please enter a valid password
    </mat-error>
    <mat-
error *ngIf="registerForm.controls.password.hasError('re
quired')">
            password is <strong>required</strong>
    </mat-error>
</mat-form-field>
</form>
```

# Exercise:

Add new input called confirm password and check if the confirm password is equal to password.

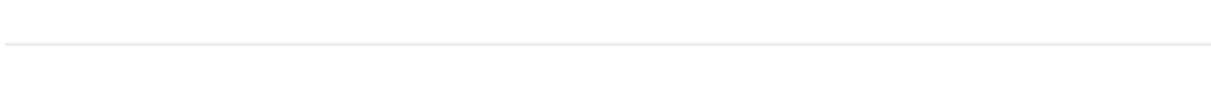# Exercise Solution:

```
<mat-form-field class="example-full-width">
 <mat-label>Confirm Password</mat-label>

        <input (change)="onchang()" matInput formControl
Name="confirmPassword" style="font-size: x-
large;" placeholder="123456">

        <mat-
error *ngIf="formRegister.controls.confirmPassword.hasEr
ror('mismatch')"> Password is not match </mat-error>

        </mat-form-field>
```

# Exercise Solution:

```
onchang(){

    if (this.formRegister.controls.password.value== this.formRegister.controls.confirmPassword.value) {

    this.formRegister.controls.confirmPassword.setErrors(null);

    }

    else{

    this.formRegister.controls.confirmPassword.setErrors({mismatch:true});

    }

}
```

# Exercise Solution:

```
 <mat-error
*ngIf="registerForm.controls.passwordConfirm.hasError('minlength') &&
!registerForm.controls.passwordConfirm.hasError('required') && registerForm.controls.passwordConfirm.value != registerForm.controls.password.value">
password and confirm password does not <strong>match</strong>

</mat-error>
```

# Exercise Solution:

```
 <mat-error *
ngIf="registerForm.controls.passwordConfirm.hasError('re
quired') " >
    Confirm password is required
</mat - error >
</mat - form - field >
```

```
<button mat-raised-button color="primary"
type="submit">Register</button>
    </form>
```

**Get value from form group:**

In Register.component.ts

```
submit(){
    const formValue = this.registerForm.value;
    console.log(formValue)

}
```

Make the button disabled if any validation is invalid :

```
<button mat-raised-button color="primary" [disabled] =
"!registerForm.valid" > Register</button >
```

## Exercise:

Add in register.component.html paragraph :" Already have an account ?
Login
navigate it to login page

# Exercise Solution:

To  go from Register page to login page you must use routing

Add in register.component.html

```
 </div >
 <p (click)="goTologinPage()" style="cursor: pointer;
text-align: center;font-size: 1.7em;margin-top: 40px;
">Already have an account ? Login here </p>
    </div >
```

# Exercise Solution:

## Add in register.component.ts

```
constructor(private route: Router)

    goToLoginPage(){
        this.route.navigate(['']);

    }
```