Angular

https://github.com/akoubayo/starter-angular.git

Prérequis

- Require Node 6.9.0 or higher
- NPM 3 or higher
- `npm install -g @angular/cli`

Start new Angular App

```
ng new PROJECT-NAME
cd PROJECT-NAME
ng serve
```

Création d'un module router

"ng generate module app-routing" => Génère automatique le module app-routing.module.ts

- Dans le app-routing.module.ts

```
import { NgModule }                                 from '@angular/core';
import { CommonModule }                             from '@angular/common';
import { RouterModule, Routes, PreloadAllModules } from '@angular/router';
import { AppModule }                                from '../app.module';

const appRoutes: Routes = [{
    path: '',

}]

@NgModule({
  declarations: [

  ],
  imports: [
    RouterModule.forRoot(
      appRoutes,
      {
        useHash: true,
        enableTracing: false, // <-- debugging purposes only
        preloadingStrategy: PreloadAllModules // To enable preloading of all lazy loaded modules
      }
    ),
  ],
  exports: [
    RouterModule,
  ],
  providers: []
})

export class AppRoutingModule { }
```

- Dans le fichier app.module.ts rajouter

```
import { AppRoutingModule }     from './app-routing/app-routing.module';
```

- Dans le fichier app.component.ts tout effacer puis rajouter

```
<router-outlet></router-outlet>
```

Création de notre première page

“ng generate component pages/login export --skip true export” => Génère automatique :

- le component login.component.ts
- le html login.html
- le css login.css
- le component login.spec.ts //Pour les tests

- Dans le fichier app-routing.module.ts ajouter

```
import { NgModule }                                 from '@angular/core';
import { CommonModule }                             from '@angular/common';
import { RouterModule, Routes, PreloadAllModules } from '@angular/router';
import { AppModule }                                from '../app.module';
import { LoginComponent }                           from './../pages/login/login.component';

const appRoutes: Routes = [{
    path: '',
    component: LoginComponent
}]

@NgModule({
  declarations: [
    LoginComponent,
  ],
  imports: [
    RouterModule.forRoot(
      appRoutes,
      {
        useHash: true,
        enableTracing: false, // <-- debugging purposes only
        preloadingStrategy: PreloadAllModules // To enable preloading of all lazy loaded modules
      }
    ),
  ],
  exports: [
    RouterModule,
  ],
  providers: []
})

export class AppRoutingModule { }
```

Création du formulaire de connexion

- Dans le fichier app-routing.module ajouter

```
import { FormsModule, ReactiveFormsModule }       from '@angular/forms'; // <= Ligne à ajouter

imports: [
   RouterModule.forRoot(
     appRoutes,
     {
       useHash: true,
       enableTracing: false, // <-- debugging purposes only
       preloadingStrategy: PreloadAllModules, // To enable preloading of all lazy loaded modules
     }
   ),
   FormsModule,                          // <= Ligne à ajouter
   ReactiveFormsModule                   // <= Ligne à ajouter
 ],
```

Création du formulaire de connexion

- Dans le fichier login.html

```
<h1>
    Page de login
</h1>
<!--
    (ngSubmit) => Action à effectué lors de l'envoie du form
    [(ngModel)]="login.email" => Bind l'entrée utilisation dans la variable login de notre component
    required  => "Champs requis pour la soumission"
    pattern=  => "Patern pour check le mail"
    [disabled]="!formulaire.form.valid" => empêche l'envoie du form tant que tout les conditions ne
    sont pas remplis
-->
<form #formulaire="ngForm" (ngSubmit)="onSubmit()">
    <label  for="email" data-error="wrong" data-success="right">Email</label>
    <input  type="email"
            name="email"
            id="email"
            [(ngModel)]="login.email"
            pattern="[a-zA-Z0-9_]+(?:\.[A-Za-z0-9!#$%&amp;'*+/=?^_`{|}~-]+)*@(?!([a-zA-Z0-9]*\.[
            a-zA-Z0-9]*\.[a-zA-Z0-9]*\.))(?:[A-Za-z0-9](?:[a-zA-Z0-9-]*[A-Za-z0-9])?\.)+[a-zA-Z0-9](?:
            [a-zA-Z0-9-]*[a-zA-Z0-9])?"
            required
    />
    <label for="password" data-error="wrong" data-success="right">password</label>
    <input type="password"
           name="passowrd"
           id="passowrd"
           [(ngModel)]="login.passowrd"
           required
           minlength="8"
    />
    <input type="submit"
           [disabled]="!formulaire.form.valid"
    />
</form>
```

(ngSubmit) => Action à effectué lors de l'envoie du form

[(ngModel)]="login.email" => Bind l'entrée utilisation dans la variable login de notre component

required => "Champs requis pour la soumission"

pattern= => "Patern pour check le mail"

[disabled]="!formulaire.form.valid" => empêche l'envoie du form tant que tout les conditions ne sont pas remplis

- Dans le fichier login.component.ts

```
import { Component, OnInit }                    from '@angular/core';
import { Router, ActivatedRoute, Params }       from '@angular/router';

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

    public login = {
        email:'',
        passowrd:''
    }

    constructor(public router: Router) { }

    ngOnInit() {
    }

    public onSubmit = () => {
        console.log(this.login)
        //this.router.navigate(['/dashboard']);
    }

}
```

Un peu de CSS avec Materialize

- Comment installer Materialize

```
npm install materialize-css --save
npm install angular2-materialize --save

npm install jquery@^2.2.4 --save
npm install hammerjs --save
```

- Editer le fichier .angular-cli.json

Dans le tableau des css ajoutez =>

```
"../node_modules/materialize-css/dist/css/materialize.css"
```

Dans le tableau des scripts ajoutez =>

```
"../node_modules/jquery/dist/jquery.js",
"../node_modules/hammerjs/hammer.js",
"../node_modules/materialize-css/dist/js/materialize.js"
```

- Dans le fichier app.module.ts ajoutez

```
import { MaterializeModule }  from 'angular2-materialize';
```

Importez MaterializeModule le dans le tableau “imports”

- Dans le fichier index.html ajoutez

```
<link href="http://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

Redémarez ng serve.

- Modification du component.html

```
<div class="col s12 white z-depth-3" style="margin-top: 25%; padding:0px 50px 0px 50px">
    <h5>
        Page de login
    </h5>
    <form materialize class="col s12" #formulaire="ngForm" (ngSubmit)="onSubmit()">
        <div class="row">
            <div class="input-field col s6">
                <input  type="email"
                        name="email"
                        id="email"
                        [(ngModel)]="login.email"
                        pattern="[a-zA-Z0-9_]+(?:\.[A-Za-z0-9!#$%&amp;'*+/=?^_`{|}~-]+)*@(?!([
                        a-zA-Z0-9]*\.[a-zA-Z0-9]*\.[a-zA-Z0-9]*\.))(?:[A-Za-z0-9](?:[a-zA-Z0-9-]*[
                        A-Za-z0-9])?\.)+[a-zA-Z0-9](?:[a-zA-Z0-9-]*[a-zA-Z0-9])?"
                        required
                        class="validate"
                />
                <label  for="email" data-error="Entrez une adresse mail valide" data-success="right">
                Email</label>
            </div>
            <div class="input-field col s6">
                <input type="password"
                       name="passowrd"
                       id="passowrd"
                       [(ngModel)]="login.passowrd"
                       required
                       minlength="8"
                       class="validate"
                />
                <label for="password" data-error="Le password doit contenir au minimum 8 caractères"
                data-success="right">password</label>
            </div>
        </div>
        <div class="row">
            <div class="input-field col s6">
                <button class="btn waves-effect waves-light"
                        type="submit"
                        name="action"
                        [disabled]="!formulaire.form.valid"
                >
                    Submit
                    <i class="material-icons right">send</i>
                </button>
```

Dans le index.html ajoutez la **class="blue-grey lighten-5"** dans la balise body
Dans app.component.ts entourez <router-outlet></router-outlet> par la balise ** </div**

Entourez le code avec
** **

** **

Dans la balise form ajoutez la directive “**Materialize**” et les **class=”col s12”**

Entourez les inputs avec :
** **
<div class=”input-field col s6>
** **
** **

Ajoutez la **class=”validate”** à vos inputs.

Modifiez les labels comme suit :
<label for="email" data-error="Entrez une adresse mail valide" data-success="right">Email</label>

Pour le bouton

```
<div class="row">
        <div class="input-field col s6">
                <button class="btn waves-effect waves-light"
                    type="submit"
                    name="action"
                    [disabled]="!formulaire.form.valid"
                >
                  Submit
                  <i class="material-icons right">send</i>
                </button>
        </div>
</div>
```

Création de la page Dashboard

Notre “dashboard“ sera composé de plusieurs component. Pour éviter à angular de tout charger dès la première fois, nous allons utiliser le lazy-module introduit par Angular 2.

Ainsi nous pourrons importez dans le module “Dashboard” les components que lui seul utilise, au lieu de les charger dans le app.module.ts.

En console :

ng generate module pages/dashboard

- le module dashboard.module.ts

ng generate component pages/dashboard

- le component dashboard.component.ts
- le html dashboard.html
- le css dashboard.css
- le component dashboard.spec.ts //Pour les tests

- Créer un fichier dashboard.routing.ts

```
import { ModuleWithProviders } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import { DashboardComponent } from './dashboard.component';

const routes: Routes = [
  { path: '', component: DashboardComponent }
];

export const routing: ModuleWithProviders = RouterModule.forChild(routes);
```

- Importez routing dans le fichier dashboard.module.ts

Ajoutons la route à notre fichier app-routing.ts

```
{    path: 'dashboard',
     loadChildren: 'app/pages/dashboard/dashboard.module#DashboardModule' },
```

Importons un module à notre dashboard

- Télécharger le module Card-profil
- Créez un dossier modules et copier le dossier Card-profils
- Dans dashboard.module : *“import { CardProfilModule } from './../../modules/card-profil/card-profil.module'”*
- Dans dasboard.html ajoutez le code suivant :

```
“<div class="row">
  <div class="col s12" style="">
    <app-card-profil></app-card-profil>
  </div>
</div>”
```

- C’est tout pour le moment (-:

Faisons communiquer nos composants

Pour le moment notre card-profil, ne sert pas à grand chose. Rendons le dynamique en lui passant transmettant des informations.

- Dans le fichier card-profil.component

```
import { Component, OnInit, Input } from '@angular/core';

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

    @Input() profil:any = {};

    constructor() {
        this.profil.first_name = "";
        this.profil.last_name  = "";
        this.profil.title      = "";
        this.profil.bullet     = 0;
    }

    ngOnInit() {
    }

}
```

- Dans le fichier card-profil.html

```
<div class="card-face__avatar">
    <!-- Bullet notification -->
    <span *ngIf="profil.bullet && profil.bullet > 0" class="card-face__bullet">{{profil.bullet}}</
    span>
    <!-- User avatar -->
    <img src="https://s3.amazonaws.com/uifaces/faces/twitter/csswizardry/128.jpg" width="110" height
    ="110" draggable="false"/>
</div>
<!-- Name -->
<h2 class="card-face__name">{{profil.first_name}} {{profil.last_name}}</h2>
<!-- Title -->
<span class="card-face__title">{{profil.title}}</span>
<!-- Cart Footer -->
```

Dans le fichier dashboard.component

```
export class DashboardComponent implements OnInit {

    public profil:any = {
        first_name:   "Damien",
        last_name:    "Altman",
        title:        "Dévellopeur chez Tilkee",
        bullet:        2
    }

    constructor() { }

    ngOnInit() {
    }

}
```

Dans le fichier dashboard.module

```
import { NgModule }                                 from '@angular/core';
import { CommonModule }                             from '@angular/common';
import { FormsModule, ReactiveFormsModule }         from '@angular/forms';
import { CardProfilModule }                         from './../../modules/card-profil/card-profil.module

import {routing}                                    from './dashboard.routing';

import { DashboardComponent }                       from './dashboard.component';


@NgModule({
  imports: [
    CommonModule,
    routing,
    CardProfilModule,
    FormsModule,
    ReactiveFormsModule
  ],
```

Dans le fichier dashboard.html

```
<div class="row">
    <div class="col s3" style="">
        <app-card-profil [(profil)]=profil></app-card-profil>
    </div>
    <div class="col s6 white z-depth-3" style="margin-top: 10px">
        <h5>Modifier son profil</h5>
            <div class="row">
            <div class="input-field col s6">
                <input  type="text"
                        name="first_name"
                        id="first_name"
                        [(ngModel)]="profil.first_name"
                        required
                        minlength="4"
                        class="validate"
                />
                <label  for="first_name" class="active"  data-error="Entrez votre prénom"
                data-success="right">First name</label>
            </div>
            <div class="input-field col s6">
                <input type="text"
                       name="last_name"
                       id="last_name"
                       [(ngModel)]="profil.last_name"
                       required
                       minlength="4"
                       class="validate"
                />
                <label for="last_name" class="active"  data-error="Entrez votre nom"
                data-success="right">Last name</label>
            </div>
            <div class="row">
                <div class="input-field col s6">
                <input type="text"
                       name="title"
                       id="title"
                       [(ngModel)]="profil.title"
                       required
                       minlength="8"
                       class="validate"
                />
                <label for="title" class="active"  data-error="Entrez votre post actuel"
                data-success="right">Title</label>
                </div>
            </div>
        </div>
    </div>
</div>

<div class="row">
        <div class="col s1">
            <a href="#">- Login</a>
        </div>
```

Testez le formulaire en modifiant les champs et appréciez le data-binding entre composants.

Maintenant cliquez sur le lien en bas de la page et revenez sur le dashboard.

Que c'est il passé, toutes les informations que j'avais modifié se sont réinitialisé. Mais comment faire pour les rendres persistantes ?

Les services

- Qu’est-ce qu’un service ?

“Ce sont des singletons, c'est-à-dire des instances uniques d'objets. Le rôle d'un service est de fournir un ensemble de tâches nécessaires au fonctionnement de votre application.”

“Lorsque nous déclarons notre service comme provider du module AppModule, ce premier est donc un singleton pour toute l'application (s'il n'est pas redéfini par ailleurs comme vous verrons par la suite…)”

Avec cette définition nous comprenons que cela va nous être utils pour notre exemple.
Commencons.

En console :

ng generate service services/user/user

- le service user.service.ts
- Les spec user.service.spec.ts // Pour les test

ng generate class services/user/user

- Une class User user.ts

```typescript
import { Injectable } from '@angular/core';
import { User }       from './user';

@Injectable()
export class UserService {

    private user:User;

    constructor() {
        this.user = new User(); // Initialisation de notre User
    }

    // Renvoie l'adresse de l'objet User
    public getUser = ():User => {
        return this.user; // C'est la référence vers l'objet user qui est renvoyer
    }

  // Renvoie une copie de l'objet User
    public getUserByCopy = ():User => {
        let user_copy:User = new User();
        for(let key in this.user) {
          user_copy[key] = this.user[key];
        }
        return user_copy;
    }

    public setUser = (user) => {
        this.user = user;
    }
}
```

```typescript
export class User {
    first_name:string    = '';
    last_name:string     = '';
    title:string         = '';
    email:string         = '';
}
```

```
import { Component, OnInit } from '@angular/core';

import { User }              from './../../services/user/user'
import { UserService }       from './../../services/user/user.service';

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

    public profil:User;

    constructor(private user_service:UserService) {

    }

    ngOnInit() {
        this.profil = this.user_service.getUser();
    }

    public setUser = (_profil:any) => {
        this.user_service.setUser(_profil);
    }

}
```

```
            <button (click)="setUser(profil)">
                save profil <!-- A rajouter dans le dashboard.html -->
            </button>
        </div>
    </div>
</div>
<div class="row">
        <div class="col s1">
            <a href="#">- Login</a>
        </div>
</div>
```

Bonus !

Nous allons créer un component pour update notre profil.

Console :

- ng generate component pages/dashboard/update

```
const routes: Routes = [
    {
        path: '',
        component: DashboardComponent,
        children: [
            {
                path: 'update',
                component:UpdateComponent
            },
        ]
    }
];
```

```
<div class="row" *ngIf="profil">
    <div class="col s3" style="">
        <app-card-profil [(profil)]=profil></app-card-profil>
    </div>
    <router-outlet></router-outlet>
</div>

<div class="row">
        <div class="col s1">
            <a href="#/dashboard/update">- Update - </a>
        </div>
</div>
```

```typescript
import { Injectable } from '@angular/core';
import { User }       from './user';
import { Subject }    from 'rxjs/Subject';
import { Observable } from 'rxjs/Observable';
@Injectable()
export class UserService {

    private user:User;
    public user_subject:Subject<any> = new Subject();

    constructor() {
        this.user = new User(); // Initialisation de notre User
    }

    // Renvoie l'adresse de l'objet User
    public getUser = ():User => {
        return this.user; // C'est la référence vers l'objet user qui est renvoyer
    }

  // Renvoie une copie de l'objet User
    public getUserByCopy = ():User => {
        let user_copy:User = new User();
        for(let key in this.user) {
          user_copy[key] = this.user[key];
        }
        return user_copy;
    }

    public setUser = (user) => {
        this.user = user;
        this.setObersableUser(this.user);
    }

    public setObersableUser(user: User) {
        this.user_subject.next(user);
    }


    public getObservableUser(): Observable<User> {
        return this.user_subject.asObservable();
    }
}
```

```
import { Component, OnInit }              from '@angular/core';
import { User }                           from './../../../services/user/user'
import { UserService }                    from './../../../services/user/user.service'
import { Router, ActivatedRoute, Params } from '@angular/router';

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

    private profil:U  class UpdateComponent

    constructor(private _user_service:UserService, private router:Router) {
    }

    ngOnInit() {
        this.profil = this._user_service.getUserByCopy();
    }

    public setUser = (_profil:any) => {
        this._user_service.setUser(_profil);
        this.router.navigate(['/dashboard']);
    }
}
```

```
<div class="col s6 white z-depth-3" style="margin-top: 10px" >
        <h5>Modifier son profil</h5>
            <div class="row">
            <div class="input-field col s6">
                <input  type="text"
                        name="first_name"
                        id="first_name"
                        [(ngModel)]="profil.first_name"
                        required
                        minlength="4"
                        class="validate"
                />
                <label  for="first_name" class="active"  data-error="Entrez votre prénom"
                data-success="right">First name</label>
            </div>
            <div class="input-field col s6">
                <input type="text"
                       name="last_name"
                       id="last_name"
                       [(ngModel)]="profil.last_name"
                       required
                       minlength="4"
                       class="validate"
                />
                <label for="last_name" class="active"  data-error="Entrez votre nom"
                data-success="right">Last name</label>
            </div>
            <div class="row">
                <div class="input-field col s6">
                <input type="text"
                       name="title"
                       id="title"
                       [(ngModel)]="profil.title"
                       required
                       minlength="8"
                       class="validate"
                />
                <label for="title" class="active"  data-error="Entrez votre post actuel"
                data-success="right">Title</label>
                </div>
            </div>
            <button (click)="setUser(profil)">
                 save profil <!-- A rajouter dans le dashboard.html -->
```

Code github

Git clone https://github.com/akoubayo/starter-angular.git

Npm install