# Loader screen

Lo scopo è quello di visualizzare un loader che si attivi automaticamente in caso vengano effettuate chiamate al back.

Per prima cosa costruisco un component che visualizzerà il loader.

loading-screen.component

la parte Html :

<div class="loading-screen-wrapper" \*ngIf="loading">

  <div class="loading-screen-icon">

    <svg class="splash-loader"

         width="65px"

         height="65px"

         viewBox="0 0 66 66"

         xmlns="http://www.w3.org/2000/svg">

      <circle class="splash-path"

              fill="none"

              stroke-width="6"

              stroke-linecap="round"

              cx="33"

              cy="33"

              r="30"></circle>

    </svg>

  </div>

</div>

la parte css :

.loading-screen-wrapper {

    position:fixed;

    display:block;

    width:100%;

    height:100%;

    top:0;

    left:0;

    background-color:rgba(74,74,74,.8);

    z-index:99999;

  }

  .loading-screen-icon {

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

  }

la parte TS :

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

import { Subscription } from 'rxjs';

import { LoadingScreenService } from 'src/app/service/screen/loading-screen.service';

@Component({

  selector: 'app-loading-screen',

  templateUrl: './loading-screen.component.html',

  styleUrls: ['./loading-screen.component.css']

})

export class LoadingScreenComponent implements OnInit, OnDestroy {

  loading: boolean = false;

  loadingSubscription: Subscription;

  constructor(private loadingScreenService: LoadingScreenService) {

  }

  ngOnInit() {

    this.loadingSubscription = this.loadingScreenService.loadingStatus.subscribe((value) => {

      this.loading = value as boolean;

    });

  }

  ngOnDestroy() {

    this.loadingSubscription.unsubscribe();

  }

}

Definito il component occorre inserire il component nell’app.module.ts nella parte declarations.

        LoadingScreenComponent

    ],

    providers: [

        httpInterceptorProviders,

Passo successivo occorre inserire il component dentro alla pagina :

app.main.component.html

come si seguito riportato :

<p-toast></p-toast>

<app-loading-screen></app-loading-screen>

Ho indicato anche il toast perché anche loro sono assoggettati alla stessa regola ossia i component vengono iniettati al posto di :

<router-outlet></router-outlet>

Indicato appunto nella pagina app.main.component.html.

A questo punto abbiamo il component LoadingScreen su tutte le pagine (component) ed è quindi pronto per essere attivato, chi lo attiva è l’interceptor seguente :

import { HttpEvent, HttpHandler, HttpInterceptor, HttpRequest } from "@angular/common/http";

import { Injectable } from "@angular/core";

import { Observable } from "rxjs";

import { finalize } from "rxjs/operators";

import { LoadingScreenService } from "../service/screen/loading-screen.service";

@Injectable()

export class LoadingScreenInterceptor implements HttpInterceptor {

  activeRequests: number = 0;

  constructor(private loadingScreenService: LoadingScreenService) {

  }

  intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {

    if (this.activeRequests === 0) {

      this.loadingScreenService.startLoading();

    }

    this.activeRequests++;

    return next.handle(request).pipe(

      finalize(() => {

        this.activeRequests--;

        if (this.activeRequests === 0) {

          this.loadingScreenService.stopLoading();

        }

      })

    )

  };

}

Ogni richiesta viene intercettata dall’interceptor che provvedere, qualora non sia già attivo, a visualizzare il loader e a spegnerlo quando tutte le richieste sono state evase.

Anche l’interceptor va indicato nell’app.module.ts come di seguito indicato :

export const httpInterceptorProviders = [

    { provide: HTTP\_INTERCEPTORS, useClass: AuthInterceptor, multi: true },

    { provide: HTTP\_INTERCEPTORS, useClass: ErrorInterceptor, multi: true },

    { provide: HTTP\_INTERCEPTORS, useClass: LoadingScreenInterceptor, multi: true }

  ];

e nell’@NgModule nella sezione providers :

    providers: [

        httpInterceptorProviders,