---creare il progetto

ng new exame-fe --no-standalone

cd /exame-fe

---installare le dipendenze principale

npm install bootstrap --save

npm install @fortawesome/fontawesome-free --save

npm install @ng-bootstrap/ng-bootstrap --save

cambiare file angular.json --linea 40

"styles": [

"src/styles.css",

"node\_modules/@fortawesome/fontawesome-free/css/fontawesome.min.css",

"node\_modules/bootstrap/dist/css/bootstrap.min.css"

],

"scripts": ["node\_modules/bootstrap/dist/js/bootstrap.bundle.min.js"],

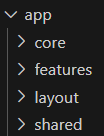
--- creare i moduli principale sotto "app"

core

features

layout

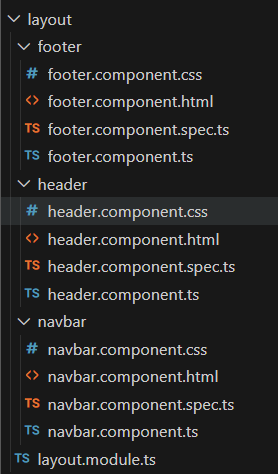
shared



--- creare il componente header

--- creare il componente footer

--- creare il componente navbar



Inserire i html per i componenti

---- cambiare il app.componente.html per utilizzare i componenti Header e Footer

<!-- src/app/app.component.html -->

<app-header></app-header>

<div class="content">

  <router-outlet></router-outlet>

</div>

<app-footer></app-footer>

Aggiornare il app.module.ts, para includere il modulo Layout

import { NgModule } from '@angular/core';

import { BrowserModule, provideClientHydration } from '@angular/platform-browser';

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

import { AppComponent } from './app.component';

import { LayoutModule } from "./layout/layout.module";

@NgModule({

  declarations: [

    AppComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    LayoutModule

],

  providers: [

    provideClientHydration()

  ],

  bootstrap: [AppComponent]

})

export class AppModule { }

provare a compilare “ng serve”

--- creare il componente features -> login

--- creare il componente features -> home

--- creare il componente features -> activity

--- creare il componente features -> timesheet

Creare all’interno del componente activity, i componenti list e row

Creare all’interno del componente timesheet, i componenti list e row

--- Creare il service auth, che sarà responsabile per le chiamate di login (new service)

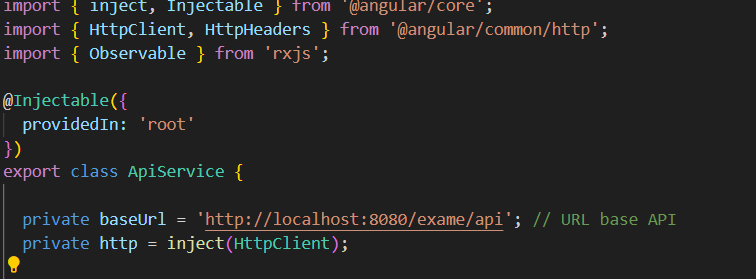
Core – auth

Copiare la classe api.service.ts, responsabile per realizzare le chiamate REST al Backend

---- verificare se il backend è running:

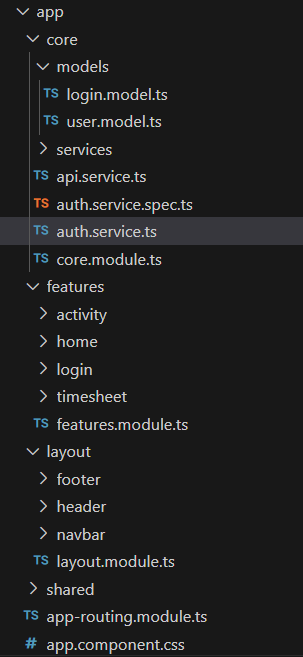
<http://localhost:8080/exame/api/openapi-ui/index.html>

cambiare il file api.service.ts e settare il base path della applicazione:



--- implementare i metodi di autenticazione, e gestione dell’utente logato.

Creare i modeli login e user come da esempi



Implementare il servizio auth come da esempi

-----------------------

Implementare la feature login, creando il html, css e login.componente.ts, chiamando il service auth.service.ts. Vedi esempi nella cartella exame.

\*\*\* Importante, che per fare funzionare correttamente i form, e potere lavorare con i dati provenuti dai modelli esistenti in tutti componenti all’interno del modulo Features, dobbiamo importare una libreria specifica “ReactiveFormsModule”:

  imports: [

    ReactiveFormsModule,

    CommonModule

  ]

})

export class FeaturesModule { }

--- implementare il componente registrazione, che verrà chiamato da un pulsante nel componente Login