Table of Contents

[Current Resources with Changes 3](#_Toc113051025)

[New Resources Added 3](#_Toc113051026)

[New Components 3](#_Toc113051027)

[Members List 3](#_Toc113051028)

[Member Detail 3](#_Toc113051029)

[Lists 3](#_Toc113051030)

[Messages 3](#_Toc113051031)

[app-routing.module – Initial Setup 3](#_Toc113051032)

[Router Outlet 4](#_Toc113051033)

[app.component.html 4](#_Toc113051034)

[/site/site-home.component.ts 4](#_Toc113051035)

[Checking routing before adding links 4](#_Toc113051036)

[Adding links - routerLink 5](#_Toc113051037)

[/site/nav/nav.component.html 5](#_Toc113051038)

[Checking routing after adding links 5](#_Toc113051039)

[Routing in Components 5](#_Toc113051040)

[/site/nav/nav.component.ts 5](#_Toc113051041)

[Toaster Notifications 6](#_Toc113051042)

[ngx-toastr 6](#_Toc113051043)

[nav.component.ts 6](#_Toc113051044)

[/site/site-home/site-home.component.ts 6](#_Toc113051045)

[/site/site-register/site-register.component.ts 6](#_Toc113051046)

[Guards 6](#_Toc113051047)

[Creating Auth Guard 6](#_Toc113051048)

[app-routing.module.ts 7](#_Toc113051049)

[Option 1: applying the guard to individual routes 7](#_Toc113051050)

[Option 2: applying the guard via dummy route [preferred] 7](#_Toc113051051)

[Group the secure links via ng-container 8](#_Toc113051052)

[/site/nav/nav.component.html 8](#_Toc113051053)

[Applying a different Theme 8](#_Toc113051054)

[Bootswatch 8](#_Toc113051055)

[angular.json 8](#_Toc113051056)

[Displaying Logged in User Name as TitleCase 9](#_Toc113051057)

[Tidying up app.module.ts 9](#_Toc113051058)

[Shared Module 9](#_Toc113051059)

[shared.module.ts 9](#_Toc113051060)

[app.module.ts 10](#_Toc113051061)

# Current Resources with Changes

1. app-routing.module.ts
2. app.component.html
3. /site/site-home/site-home.component.ts
4. /site/site-home/site-home.component.ts
5. /site/site-register/site-register.component.ts
6. /site/nav/nav.component.html
7. /site/nav/nav.component.ts

# New Resources Added

1. /site/members/member-list
2. /site/members/member-detail
3. /site/lists
4. /site/messages
5. /core/guards/auth

# New Components

## Members List

Create a new component with following command

* ng g c /site/members/member-list --skip-tests

## Member Detail

Create a new component with following command

* ng g c /site/members/member-detail --skip-tests

## Lists

Create a new component with following command

* ng g c /site/lists --skip-tests

## Messages

Create a new component with following command

* ng g c /site/messages --skip-tests

# app-routing.module – Initial Setup

* This is the module where routes will be created to get to different resources of the site. (Clean up will be done later).
* This is imported in the app.module.ts in the imports array.

import { SiteHomeComponent } from './site/site-home/site-home.component';

import { MemberListComponent } from './site/members/member-list/member-list.component';

import { MemberDetailComponent } from './site/members/member-detail/member-detail.component';

import { ListsComponent } from './site/lists/lists.component';

import { MessagesComponent } from './site/messages/messages.component';

//add the components here

//first empty one is the default route

//\*\* is the default route, it can be any component

const routes: Routes = [

  { path: '', component: SiteHomeComponent },

  { path: 'members/list', component: MemberListComponent },

  { path: 'members/detail/:id', component: MemberDetailComponent },

  { path: 'lists', component: ListsComponent },

  { path: 'messages', component: MessagesComponent },

  { path: '\*\*', component:  SiteHomeComponent, pathMatch: 'full'} //path match full is important here

];

# Router Outlet

## app.component.html

Add the router outlet to the app.component.html.

    <!--add the router outlet-->

    <!--<app-site-home [title]="title"></app-site-home>-->

    <router-outlet></router-outlet>

This has created an issue since we are passing the title to the site home component.

### /site/site-home.component.ts

Change the @Input() title to just title since this is not being received any more

  //after route-outlet this is not happening any moree

  //@Input() title = '';

  //fill the title in this component now

  title = '';

and then in ngOnInit fill it from environment

this.title = environment.title;

## Checking routing before adding links

Build the angular app again

* ng serve

Go to <https://localhost:4200/> and you should be able to navigate to it.

Also try the following and you should be able to get to these

* <https://localhost:4200/members/list>
* <https://localhost:4200/members/detail/2>
* <https://localhost:4200/lists>
* <https://localhost:4200/messages>

Finally try a route that doesn’t exist and you’ll be redirected to the home page due to the wildcard route.

## Adding links - routerLink

### /site/nav/nav.component.html

Open the /site/nav/nav.component.html and change the links accordingly.

#### Site Title

##### From

<a class="navbar-brand" href="#"><span class="title">{{title}}</span></a>

##### To

<a class="navbar-brand" routerLink="/"><span class="title">{{title}}</span></a>

#### Matches Link

##### From

<a class="nav-link" aria-current="page" routerLink="/" (click)="onNavBarItemClickCloseNavBar()">Matches</a>

##### To

<a class="nav-link" aria-current="page" routerLink="/members/list" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Matches</a>

#### Lists Link

##### From

<a class="nav-link" routerLink="/" (click)="onNavBarItemClickCloseNavBar()">Lists</a>

##### To

<a class="nav-link" routerLink="/lists"" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Lists</a>

#### Messages

##### From

<a class="nav-link" routerLink="/" (click)="onNavBarItemClickCloseNavBar()">Messages</a>

##### To

<a class="nav-link" routerLink="/messages"" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Messages</a>

## Checking routing after adding links

Click the links and you should be able to navigate to these.

## Routing in Components

### /site/nav/nav.component.ts

* On logout redirect the user to the site home page and on login redirect to members/list.
* First inject angular Router via constructor

private router: Router

#### Logout

Add following to the onLogout method, towards the end of the code.

    //redirect the home page

    this.router.navigateByUrl('/');

#### Login

Add following to the onLogin method, towards the end of the success part.

        //redirect to the /members/list

        this.router.navigateByUrl('/members/list');

# Toaster Notifications

## ngx-toastr

Add package ngx-toastr. Since this project got built with Angular 13.0.2, make sure to install appropriate version of ngx-toastr.

* npm install [ngx-toastr@14.3.0](mailto:ngx-toastr@14.3.0)

Then go to angular.json and add the toastr.css to the styles array

* ./node\_modules/ngx-toastr/toastr.css

Finally add the toaster module to the app.module.ts

import { ToastrModule } from 'ngx-toastr';

And then to import array add

ToastrModule.forRoot({positionClass: 'toast-bottom-right'})

## nav.component.ts

change the alert to display the error message via the toastr notification.

First inject the ToastrService

private toastrService: ToastrService

and then use it inside the displayError method

    //alert(`displayError-${from} Error: ${this.error}`);

    this.toastrService.error(this.error);

## /site/site-home/site-home.component.ts

Do the same as above and in displayError method comment the alert and instead use toastr notification.

## /site/site-register/site-register.component.ts

Do the same as above and in displayError method comment the alert and instead use toastr notification.

# Guards

## Creating Auth Guard

Create the auth guard in /core/guards folder with the following command

* ng g guard /core/guards.auth --skip-tests

This will present the options to select from. Select **CanActivate** and hot enter.

Graphical user interface, application

Description automatically generated

Change the code as per following

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

import { ActivatedRouteSnapshot, CanActivate, RouterStateSnapshot, UrlTree } from '@angular/router';

import { map, Observable } from 'rxjs';

import { ToastrService } from 'ngx-toastr';

import { AccountService } from '../services/account.service';

import { UserTokenDto } from '../models/userTokenDto.model';

@Injectable({

  providedIn: 'root'

})

export class AuthGuard implements CanActivate {

  constructor(private accountService: AccountService, private toastrService: ToastrService) { }

  canActivate(): Observable<boolean>  {

    //account service already has the observable that we will use. If it is null then the user is not logged in

    return this.accountService.currentUser$.pipe(

      map((user: UserTokenDto) => {

        if (user)

          return true;

        //in real time we do not do this.

        this.toastrService.error("You are not authorized to view the resource. Login please!");

        return false;

      })

    );

  }

}

## app-routing.module.ts

Next open the app-routing.module and apply the auth guard to the secure resources.

### Option 1: applying the guard to individual routes

const routes: Routes = [

  { path: '', component: SiteHomeComponent },

  { path: 'members/list', component: MemberListComponent, canActivate: [AuthGuard] },

  { path: 'members/detail/:id', component: MemberDetailComponent, canActivate: [AuthGuard] },

  { path: 'lists', component: ListsComponent, canActivate: [AuthGuard] },

  { path: 'messages', component: MessagesComponent, canActivate: [AuthGuard] },

  { path: '\*\*', component:  SiteHomeComponent, pathMatch: 'full'} //path match full is important here

];

### Option 2: applying the guard via dummy route [preferred]

const routes: Routes = [

  { path: '', component: SiteHomeComponent },

  //dummy route to group secure resources together

  {

    path: '',

    runGuardsAndResolvers: 'always',

    canActivate: [AuthGuard],

    children: [

      { path: 'members/list', component: MemberListComponent },

      { path: 'members/detail/:id', component: MemberDetailComponent },

      { path: 'lists', component: ListsComponent },

      { path: 'messages', component: MessagesComponent },

    ]

  },

  { path: '\*\*', component:  SiteHomeComponent, pathMatch: 'full'} //path match full is important here

];

## Group the secure links via ng-container

### /site/nav/nav.component.html

<!--only show links when logged in-->

                <ul class="navbar-nav me-auto mb-2 mb-md-0">

                    <ng-container \*ngIf="isLoggedIn">

                        <li class="nav-item">

                            <a class="nav-link" aria-current="page" routerLink="/members/list" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Matches</a>

                        </li>

                        <li class="nav-item">

                            <a class="nav-link" routerLink="/lists" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Lists</a>

                        </li>

                        <li class="nav-item">

                            <a class="nav-link" routerLink="/messages" routerLinkActive="active" (click)="onNavBarItemClickCloseNavBar()">Messages</a>

                        </li>

                    </ng-container>

                </ul>

# Applying a different Theme

## Bootswatch

Different themes will get applied via bootswatch. Install it using npm.

* npm install bootswatch

Go to <https://bootswatch.com/> to view different themes and then apply which ever you like.

## angular.json

A different theme could be applied via styles array. Cosmo theme is applied below.

            "styles": [

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

              "./node\_modules/ngx-bootstrap/datepicker/bs-datepicker.css",

              "./node\_modules/bootswatch/dist/cosmo/bootstrap.css",

              "./node\_modules/font-awesome/css/font-awesome.css",

              "./node\_modules/ngx-toastr/toastr.css",

              "src/styles.css"

            ],

# Displaying Logged in User Name as TitleCase

Angular has a built in pipe to use and it is titlecase.

Go to nav.component.html and apply the titlecase pipe.

<button type="button" class="btn btn-info">Welcome {{userInfo.userName | titlecase}}</button>

# Tidying up app.module.ts

## Shared Module

Create a new shared module in /core/modules folder using the following command

* ng g m /core/modules/shared –flat

Then open the app.module.ts and shared.module.ts and move items from app module to shared module.

Once done, add SharedModule to app module imports array.

### shared.module.ts

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

import { CommonModule } from '@angular/common';

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import { HttpClientModule } from '@angular/common/http';

import { FormsModule } from '@angular/forms';

import { BsDropdownModule } from 'ngx-bootstrap/dropdown';

import { ToastrModule } from 'ngx-toastr';

import { SiteHomeComponent } from '../../site/site-home/site-home.component';

import { SiteRegisterComponent } from '../../site/site-register/site-register.component';

import { MemberListComponent } from '../../site/members/member-list/member-list.component';

import { MemberDetailComponent } from '../../site/members/member-detail/member-detail.component';

import { ListsComponent } from '../../site/lists/lists.component';

import { MessagesComponent } from '../../site/messages/messages.component';

//what ever is imported here, the same should be exported as well

@NgModule({

  declarations: [

    SiteHomeComponent,

    SiteRegisterComponent,

    MemberListComponent,

    MemberDetailComponent,

    ListsComponent,

    MessagesComponent

  ],

  imports: [

    CommonModule,

    BrowserAnimationsModule,

    HttpClientModule,

    FormsModule,

    BsDropdownModule.forRoot(),

    ToastrModule.forRoot({positionClass: 'toast-bottom-right'})

  ],

  exports: [

    BrowserAnimationsModule,

    HttpClientModule,

    FormsModule,

    BsDropdownModule,

    ToastrModule

  ]

})

export class SharedModule { }

### app.module.ts

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

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

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

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

import { NavComponent } from './site/nav/nav.component';

import { SharedModule } from './core/modules/shared.module';

@NgModule({

  declarations: [

    AppComponent,

    NavComponent,

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    SharedModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }