**ROUTE GUARDS**

1. In this exercise you will use route guards and child routes, among other things.
2. Create a new application with routing. Copy starter files into it.
3. In src\app\app-routing.module.ts, add paths and components for a default route (no url), home, about and admin.
4. Add an error page and add it to the routing setup so that it’s displayed whenever a non-existent path (non-valid route) is added in the address bar.
5. First try typing in a path you didn’t route to a component, and look at the console.log.
6. We will route it to error/route.error.component.ts
7. Import the component in the routing module like the other components. Test by typing in a bad url.
8. Right now anyone using the site can access the Admin page. In this part of the exercise we will add a guard to the Admin route that requires users to “log in” before they can access it. It involves the following, all coming up next:
   1. Add a Login button and an indicator for logged in status.
   2. Add a guard service that implements "CanActivate".
   3. Add a login() function to the component.
   4. Add the guard to the route in the route table.
9. In src/app/app.component.html:
   1. Add this after the end of the button end div and the starting main tag.

<div class=login >

<input class=user type=button (click)="login()" value="{{loginLabel}}">

<span class=user [hidden]="user==='na'">User: {{user}}</span>

</div>

1. Add the route guard to auth.service.ts:
2. This file contains a service that implements the *canActivate* guard and manages the *isLoggedIn* state variable:

export class AuthService implements CanActivate { isLoggedIn: boolean = false;

login(){

this.isLoggedIn = true; console.log('AuthService: logging in');

}

logout(){

this.isLoggedIn = false; console.log('AuthService: logging out');

}

constructor() {} canActivate() {

return this.isLoggedIn;

}

}

1. To add a login() function to the component in src/app/app.component.ts
   1. Import the Auth service
   2. Inject the AuthService in the constructor:

constructor( private auth: AuthService, private router: Router ){}

* 1. Add a login method to the AppComponent class:

**login(){**

**if(this.loginLabel === 'Log in'){ this.auth.login();**

**this.user = 'Admin'; this.loginLabel = 'Log out';**

**}else{**

**this.auth.logout(); this.user = 'na'; this.loginLabel = 'Log in'; let link = ['/home']; this.router.navigate(link);**

**}**

**}**

* 1. Save and close the file.

1. Add the guard to the route table in src/app/app-routing.module.ts
2. Import the AuthService. Add the canActivate guard to the admin route in the appRoutes array. Hint:

{path: 'admin', component: AdminComponent**, canActivate: [AuthService]** }

1. Save and close the file.
2. Test the guarded route by logging in and out (just clicking the button).