In this thread I would like to discuss route guards. What a route guard is will be defined and why they are used will also be described. There are functions in Angular that are a part of route guards. Which ones are available, how they work, and under what circumstances they should be used will also be a topic of conversation. So, what is a route guard?

Route guards are similar in nature to using Authentication and Authorization on the server side when we check a user’s access and permissions to be able to access certain areas of our sites. To do this in Angular, we use Route Guards to “prevent users to access a certain area of the application that is not permitted or has access to.” (Shah, n.d.)

To be able to accomplish this, there are four types of Route Guards that we can employ. The route guards that we can use are listed below:

* canActivate – “Interface that a class can implement to be a guard deciding if a route can be activated. If all guards return true, navigation continues. If any guard returns false, navigation is cancelled. If any guard returns a [UrlTree](https://angular.io/api/router/UrlTree), the current navigation is cancelled and a new navigation begins to the [UrlTree](https://angular.io/api/router/UrlTree) returned from the guard.” (*Angular*, n.d.)
* canActivateChild – “Interface that a class can implement to be a guard deciding if a child route can be activated. If all guards return true, navigation continues. If any guard returns false, navigation is cancelled. If any guard returns a [UrlTree](https://angular.io/api/router/UrlTree), current navigation is cancelled and a new navigation begins to the [UrlTree](https://angular.io/api/router/UrlTree) returned from the guard.” (*Angular*, n.d.-b)
* canDeactivate – “Interface that a class can implement to be a guard deciding if a route can be deactivated. If all guards return true, navigation continues. If any guard returns false, navigation is cancelled. If any guard returns a [UrlTree](https://angular.io/api/router/UrlTree), current navigation is cancelled and a new navigation begins to the [UrlTree](https://angular.io/api/router/UrlTree) returned from the guard.” (*Angular*, n.d.-c)
* canLoad – “Interface that a class can implement to be a guard deciding if children can be loaded. If all guards return true, navigation continues. If any guard returns false, navigation is cancelled. If any guard returns a [UrlTree](https://angular.io/api/router/UrlTree), current navigation is cancelled and a new navigation starts to the [UrlTree](https://angular.io/api/router/UrlTree) returned from the guard.” (*Angular*, n.d.-d)

“CanActivate basically answers the question: “Does the user have access to this route?” We use this guard to prevent access to users who are not authorized to access a route.” (Victor, 2022) To really understand when to use canActivateChild lets consider the fact that we may want to give access to someone who can access the route in the parent component, yet we want to restrict them from accessing the routes in the children components, we can invoke the canActivateChild on all or some of the routes depending on what the user needs to see. Alternatively, we could put canActivate guards on each child route, but that could be tedious. “The canDeactivate guard can be used in the scenario where a user is changing form data and before saving, user tries to navigate away.” (*Angular CanDeactivate Guard Example*, n.d.) “The canLoad Guard prevents the loading of the lazy loaded module. We generally use this guard when we do not want to unauthorized user to navigate to any of the routes of the module and stop them from even seeing the source code of the module.” (*Angular CanLoad Guard Example*, 2020)

In conclusion we now have a basic understanding of how to protect routes in Angular with route guard. We now know that there are 4 ways to do this. Briefly summarized, each route guard has the capability to decide whether access is needed for a particular route for users to be able to successfully navigate a site that we build in Angular.

References:

Shah, K. (n.d.). *Angular - Route Guard*. Retrieved January 16, 2023, from <https://www.c-sharpcorner.com/article/angular-route-guard/>

*Angular*. (n.d.). Retrieved January 16, 2023, from <https://angular.io/api/router/CanActivate>

*Angular*. (n.d.-b). Retrieved January 16, 2023, from <https://angular.io/api/router/CanActivateChild>

*Angular*. (n.d.-c). Retrieved January 16, 2023, from <https://angular.io/api/router/CanDeactivate>

*Angular*. (n.d.-d). Retrieved January 16, 2023, from <https://angular.io/api/router/CanLoad>

Victor, N. L. (2022, March 8). *Angular Basics: CanActivate—Introduction to Routing Guards*. Telerik Blogs. Retrieved January 16, 2023, from <https://www.telerik.com/blogs/angular-basics-canactivate-introduction-routing-guards>

*Angular CanDeactivate Guard Example*. (n.d.). Retrieved January 16, 2023, from <https://www.concretepage.com/angular-2/angular-candeactivate-guard-example>

*Angular CanLoad Guard Example*. (2020, April 21). TekTutorialsHub. Retrieved January 16, 2023, from <https://www.tektutorialshub.com/angular/angular-canload-guard-example/>