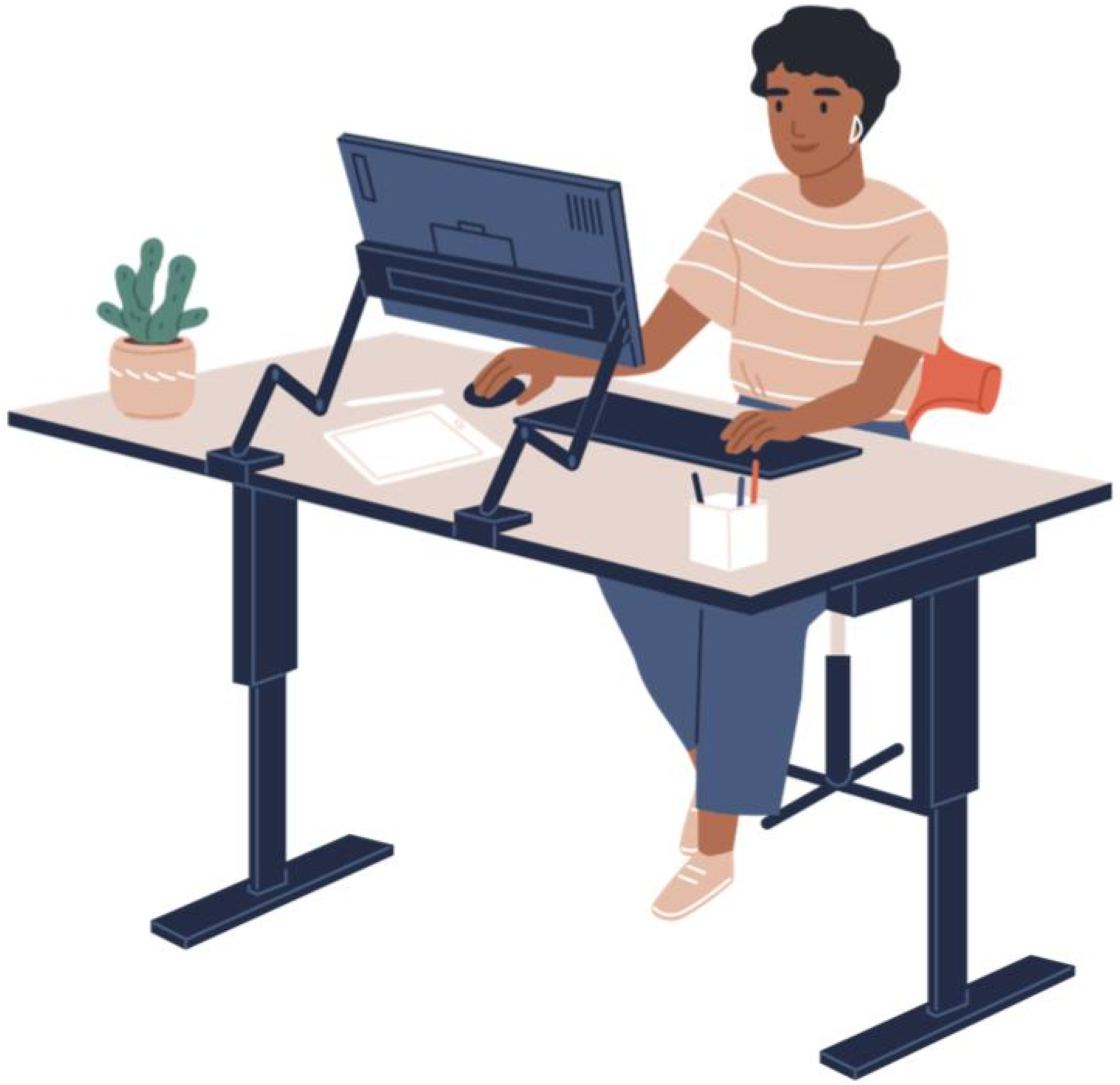
Learning Consolidation **Guard Routes in an SPA**



Menu

00:01:00:00:12 01/ 12

In this Sprint, you learned to…

- Control access to application parts using route guards
- Configure routes to add a route guard
- Add a guard to control route activation
- Add a guard to control route deactivation

Control Access With Navigation

- Routing in an SPA allows a user to navigate and access different parts of the application.
- The navigation to other parts should be controlled.
- The reasons for doing so could be:
  - To protect parts of the application from unauthenticated or unauthorized access.
  - To ensure data is available before a part of an application is accessed.
  - To allow users to save the unsaved changes before leaving a component.
  - To request confirmation from users before leaving the component.

Angular Route Guards

- Angular allows you to control access to different parts of an application through guards.
- Guards are added to the route configuration in the router module.
- The return value from guards helps in controlling the route's behavior.
- If the returned value is:
  - True, the navigation process continues.
  - False, the navigation process stops, and the user stays on the current view.
  - `UrlTree`, the current navigation is cancelled, and the new navigation process is initiated twith the `UrlTree` returned.
- The return value can be synchronous or asynchronous.
- For asynchronous returns, the guard methods return either `Observable<boolean>` or `Promise<boolean>`.

Types of Route Guards

- Guards in Angular are injectable classes that implement guard interfaces.
- The router supports multiple guard interfaces:
    - `CanActivate`: Mediate navigation to a route.
    - `CanActivateChild`: Mediate navigation to a child route.
    - `CanDeactivate`: Mediate navigation away from the current route.
    - `Resolve`: Perform route data retrieval before the route activation.
    - `CanLoad`: Mediate navigation to a feature module loaded asynchronously.

Route Guard Execution Order

- Multiple guards can exist at every level of the routing hierarchy.
- The order of execution is as follows:
  - `CanDeactivate`:
    - The Router checks these guards first, from the deepest child route to the top.
  - `CanActivate` and `CanActivateChild`:
    - The Router checks these from the top down to the deepest child route.
  - `CanLoad`:
    - If the feature module is loaded asynchronously, the router checks for this guard before the module is loaded.
- If any of the guards return false, the execution of pending guards is cancelled, and the entire navigation is cancelled.

Quick Check: Scenario

myBlog is a blogging site where interested bloggers can post their blogs. To blog, the bloggers must first register on the site and ensure that they are logged in while posting the blogs.

The blogs can receive comments from other registered users.
The blog post and its comments need to be posted before leaving the view.

Quick Check: Scenario Explained

- Here are the routes and the mapped components that allow users of the myBlog website to navigate to various parts of the application.

Route	Component	Description
\blogs	`BlogListComponent`	Displays list of all the blogs.
\blogs\:blogId	`BlogEditComponent`	Displays details of the selected blog for editing, only if the same is posted by the logged-in user.
\blogs\:blogId\comments	`CommentListComponent`	Displays list of comments posted on selected blog.
\blogs\:blogId\comments\:commentId	`CommentEditComponent`	Displays details of the selected comment for editing, only if the same is posted by the logged-in user.
\blogs\post-blog	`PostBlogComponent`	Allows logged-in users to post new blogs.
\blogs\:blogId\comments\post-comment	`PostCommentComponent`	Allows logged-in users to add new comments to the blogs.

Quick Check

Identify the route guards that can be applied to these components from among the following options:

1. `CanActivate`
2. `CanActivateChild`
3. `CanDeactivate`
4. `CanLoad`

Quick Check

Enter your answers in the Guards column.

Route	Component	Guards
\blogs	`BlogListComponent`	
\blogs\:blogId	`BlogEditComponent`	
\blogs\:blogId\comments	`CommentListComponent`	
\blogs\:blogId\comments\:commentId	`CommentEditComponent`	
\blogs\post-blog	`PostBlogComponent`	
\blogs\:blogId\comments\post-comment	`PostCommentComponent`	

Menu

00:10 00:00:12 10/ 12

Quick Check: Solution

Route	Component	Guards
\blogs	`BlogListComponent`	**None**
\blogs\:blogId	`BlogEditComponent`	**`CanActivate`**
\blogs\:blogId\comments	`CommentListComponent`	**None**
\blogs\:blogId\comments\:commentId	`CommentEditComponent`	**`CanActivate` – allows only logged-in users to navigate here** **`CanDeactivate` – ensures that the user saves changes before leaving**
\blogs\post-blog	`PostBlogComponent`	**`CanActivate`**
\blogs\:blogId\comments\post-comment	`PostCommentComponent`	**`CanActivate` – allows only logged-in users to navigate here** **`CanDeactivate` – ensures that the user saves changes before leaving**

Menu

00:11:00:00:12 11/ 12