**Discussion Assignment 3.1 - Route Guards**

Yakut Ahmedin

Bellevue University

WEB425 Angular with TypeScript

Professor Richard Krasso

Jun 08, 2023

**Route Guards**

When it comes to building Angular applications, one of the key aspects to consider is proper access control and navigation management. This is where Angular route guards step in to help. Route guards serve as gatekeepers, enabling us to control and secure access to different routes within our application. They determine whether a user is authorized to access a specific route or perform a particular action.

There are several reasons why we use route guards in our applications. Firstly, they allow us to enforce authentication and authorization rules. By implementing route guards, we can ensure that only authenticated users with the necessary privileges can access certain routes or perform specific actions. This enhances the overall security and integrity of our application by preventing unauthorized access to sensitive areas.

Secondly, route guards enable us to handle navigation-related operations effectively. For instance, we can use route guards to prompt users with confirmation dialogs before leaving a page or to check for unsaved changes and provide warnings. This ensures a smooth and controlled user experience, minimizing the risk of accidental data loss or inconsistent application states.

In Angular, there are different types of route guards that are processed in a specific order. These include canDeactivate, canLoad, canActivateChild, canActivate, and resolve. Each type serves a specific purpose and can be used in various scenarios.

The canActivate guard is commonly used to limit route access to specific users and ensure prerequisites are met. canActivateChild restricts access to child routes and ensures prerequisites for the child route are fulfilled. canDeactivate checks criteria before leaving a route, such as unsaved changes or incomplete operations, and provides alerts or confirmation prompts to the user.

The canLoad guard prevents the asynchronous loading of modules or components if the user is not authorized, optimizing the app's performance by loading only necessary resources. Lastly, the resolve guard prefetches data before activating a route, rendering the component with the required data. It is useful when displaying the component with preloaded data rather than an empty state initially

**References**

F. (2022, February 4). Route Guards — Angular. Medium. <https://levelup.gitconnected.com/route-guards-angular-1ea6e596ce65>

Laforge, T. (2023, April 28). Everything you need to know about route Guard in Angular. Medium. <https://itnext.io/everything-you-need-to-know-about-route-guard-in-angular-697a062d3198>