Zahava Gopin

WEB 425: discussion 3.1

6/5/2023

Angular Route Gaurds

Angular route guards are a powerful feature provided by the Angular framework that allow developers to control the navigation and access to different routes within an Angular application. Route guards act as checkpoints or filters that are placed on specific routes to determine whether a user is allowed to navigate to a particular route or not. They provide a way to protect routes, enforce authentication, and implement authorization logic.

The primary purpose of route guards is to ensure that the application's routes are accessed by authorized users only. They help in implementing security measures and preventing unauthorized access to sensitive routes or data. By using route guards, developers can define conditions and rules that need to be satisfied before a user can navigate to a particular route. This can include checking if the user is authenticated, verifying user roles or permissions, or performing any other custom logic to determine access.

In Angular, the most used route guard function is canActivate. It is used to determine if a route can be activated or not. The canActivate function returns a boolean value indicating whether the navigation should be allowed or not. If it returns true, the navigation proceeds, and if it returns false, the navigation is blocked. Developers can implement custom logic inside the canActivate function to perform checks like checking authentication status or verifying user roles.

Besides canActivate, Angular also provides other route guard functions, including:

canActivateChild: This guard is used to determine if the child routes of a particular route can be activated or not. It is useful when you want to protect all child routes under a parent route.

canDeactivate: This guard is used to determine if a route can be deactivated or not. It allows you to perform checks before leaving a particular route. It is commonly used for implementing dirty checking, confirmation dialogs, or preventing unsaved changes from being discarded.

canLoad: This guard is used to determine if a lazy-loaded module can be loaded or not. It is useful when you want to prevent loading a module if certain conditions are not met, such as checking if the user is authenticated or has the necessary permissions (Sharma, 2019).

Route guards should be used in situations where it is essential to restrict access to certain routes or modules based on specific conditions.

Resources:

Chenkie, R. (2019, July 9). *Angular Authentication: Using Route Guards*. Medium. <https://medium.com/@ryanchenkie_40935/angular-authentication-using-route-guards-bf7a4ca13ae3>

Sharma, P. (2019, December). *Understanding Angular Guards - codeburst*. Medium; codeburst. https://codeburst.io/understanding-angular-guards-347b452e1892

‌