Route Guards

What is Route Guards ?

They are interface provided by angular which when implemented allow us to control the accessibility of route based on condition provided in class implementation of that Interface

Five types of Route Guards

- ★ Can Activate
- ★ Can Activate Child
- ★ Can Load
- ★ Can Deactivate
- ★ Resolve

1. Can Activate → Can Activate

Auth Guard Service
(define class AuthGuard
Implement canActivate Method)

will call everytime user navigate to this field

{path : 'user/:id', Component : HomeComponent
Can Activate : [AuthGuard]}

// Auth Guard Service

canActivate (route : ActivatedRouteSnapshot,
state : RouteStateSnapshot) :
Observable <boolean> | Promise <boolean> {

{
return true; (If navigation happen)
}

Now Use this service

2. CanActivateChild – Similar to canActivate but also prevent access to child routes of given Route.

3. CanLoad – In angular we Implement lazy loading to download only required modules using loadchildren.

If we want to prevent unauthorized user we can use CanActivate Guard.

that will do job but also download Module.

Now to control navigation as well as prevent downloading of that module we can use CanLoad Guard

canLoad (Route: Route, segments: UrlSegment[])
: Observable <boolean> | Promise <boolean>

{
  return true; (whether want to download module or not.)
}

4. canDeactivate – This guard is useful when you want to prevent user from accidentally navigating away without saving or some undone task.

canDeactivate (component: T, currentRoute: ActivatedRouteSnapshot, currentState: RouterStateSnapshot, nextState?):
Observable <boolean | UrlTree> | Promise <boolean | UrlTree> | boolean | UrlTree;

Here T is interface, that we will be using in component as well as guard

Page
Date

// CanDeactivate Service

→ export interface canComponentDeactivate {

canDeactivate : () ⇒ Observable <boolean>
| Promise <boolean> | boolean;

}

→ export class CanDeactivate implements
CanDeactivate < canDeactivate Component Deactivate>

{
CanDeactivate (component : CanComponentDeact
, currenRoute , current state, next state)

{ return component. canDeactivate();

}

}

Now Implement canComponentDeactivate
interface in component.

→ export class EditComponent implements OnInit,
CanComponentDeactivate {

canDeactivate (): Observable <boolean>
| Promise <boolean> {

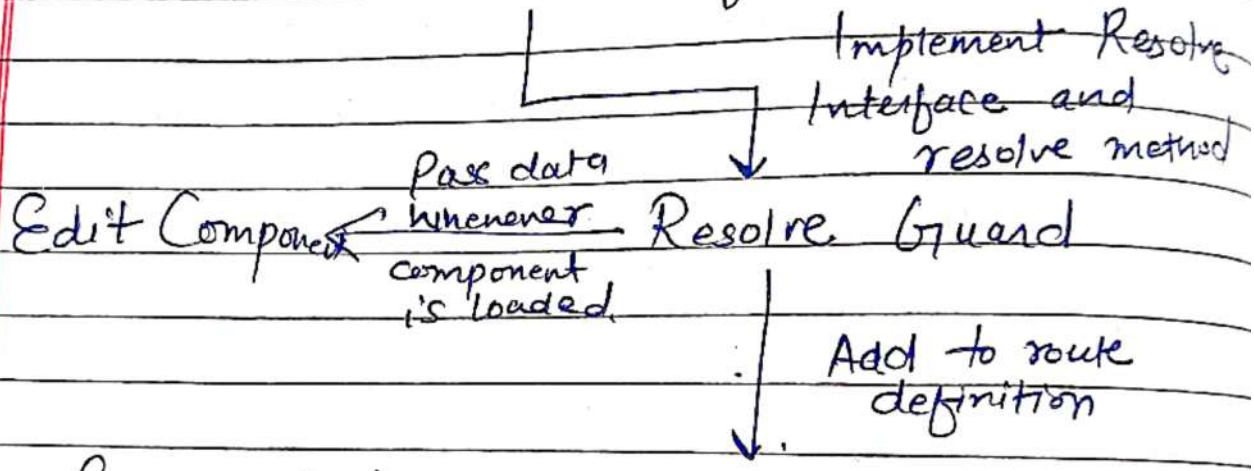
}

}

5 Resolve Guard

While data communication b/w components, Sometimes data is so heavy that is not possible to pass data through query params.

To handle this we have Resolve Guard.




```
{ path: 'id', component: EditComponent,
    resolve: { data: ResolveGuard } }

// Resolve Guard Service

resolve (route, state)
    {
        return this.fetchService.get (+ rout.
                                        params['id']
    }
```