# Instructions – Assignment 3.4 – Guarding Routes

profile-app, part 4

**Instructions**

* Make a copy of the secure-profile-app from Exercise 3.3 and add it to your week-3 directory
* Rename the application to enhanced-secure-profile-app
* Delete the node\_modules directory
* Delete the package-lock.json file
* Open the angular.json file and find and replace all “secure-profile-app” entries with “enhanced-secure-profile-app”
* Open the package.json file and change the name to “enhanced-secure-profile-app”
* Run npm install and ng serve
  + You are doing this to test the application and confirm there are no errors
* Generate a new Angular guard and name it sign-in
  + ng g g sign-in
    - Select the default “CanActivate” option
* sign-in.guard.ts
  + Add an import statement for Angular’s built-in Router
    - Import { Router } from ‘@angular/router’;
  + Add the Router to the guards constructor (note: you will need to create a constructor)
    - constructor(private route: Router)
  + In the body of the canActivate() function add a variable for isLoggedIn and assign it the returned value from the queryParams function
    - let isLoggedIn = next.queryParams.isLoggedIn
  + If isLoggedIn equals true, return true
  + If isLoggedIn equals false, navigate users back to the sign-in page
    - this.router.navigate([‘/]);
* home.component.ts
  + Remove the isLoggedIn variable and the variable assignment code from the components constructor
* home.component.html
  + Remove the \*ngIf statement and the #showError <ng-template>
* app.routing.ts
  + Add an import statement for the SignInGuard
    - import { SigninGuard } from ‘./sign-in.guard’;
  + Under the /home path, add an entry for “canActivate” and passin-in the SignInGuard
    - canActivate: [SignInGuard]
* Run and test the application
  + First test the application by selecting the Submit button
  + Next, test the application by changing the browsers URL to <http://localhost:4200/home>
    - Notice in the second test the application automatically redirects users to the sign-in component.  This is because we now have a Guard in place protecting unauthorized users from accessing the HomeComponent
    - These are the building blocks to enforcing authorization (what users can see in your application).  In WEB 450 we expand on this principle by incorporating a database