# Test task for a senior web developer candidate

## Application structure

It should be an SPA built up with AngularJS as a framework and backed by MS SQL database. The application should contain 3 separate modules each of which has its own set of backend and client-side services. The app consists of 2 screens: home screen and management screen. Home screen should support 2 modes: for authenticated and non-authenticated user. Management screen can only be accessed by logged in users. Each screen has a menu with the next items:

* Link to home page
* Link to management screen
* Login block which displays:
  + For authenticated user: current user name and dropdown submenu with single item “Logout”, or
  + For anonymous - “Login” link and dropdown form to login.

There are permissions that allow user to **see**, **edit** or **prohibit using** of a module. The permission setting per user per module can only have one value. If a user has not appropriate permissions for a module it should not only see the visual part, but also should not access the module at all (neither scripts nor any URL of the module should be accessible, as a plus the whole angular module should not be loaded into the memory).

When the App starts for the first time the Administrator user should be created. Administrator has maximum permissions for all modules (‘edit’ permissions).

## Pages description:

* **Home page** contains only “welcome” module with greetings message for an authenticated user. If a user has not been authenticated the page should contain some logo (not within a module, maybe as a background or so). If the “welcome” module is not accessible at all (has “prohibit using” permission setting) the page shows only menu and empty space on the module’s place.
* **Management screen** consists as a maximum of two modules: “user management” and “module list”. If both of these modules have set up as “prohibit using” for current user, the main menu shouldn’t contain the link to the management screen.

## Modules description:

* “**Welcome**” module only displays some greetings message for the currently logged in user. It could, for example, write some user properties: name or last login time.
* “**User management**” module contains [master-detail KendoUI grid](http://demos.telerik.com/kendo-ui/grid/detailtemplate) of users in the system with the next columns: first name, last name, and comma separated list of modules available for the user. When user expands selected row the Edit form appears. It is possible to change user’s name/surname and the list of available modules. If the module is available it is possible to set up whether the module can be edited or not.

If user opens management screen and it has only see permission for “user management” module the grid should not contain details row.

* “**Module list**” module contains grid (it does not necessarily have to be KendoUI grid, it might be an Angular UI grid or some other else) with the list of modules available for the current user with two columns (module’s name and the name of the user who has granted permissions to this module for the current user). Grid should support inline editing. User can rename module as it wants to see it. This renaming should not affect other users in the system. When user renames a module, the name of a module in user management grid should be changed too automatically (both in master and detail rows)

## Technologies to be used:

* WebApi 2
* Entity Framework 6
* AngularJS 1.3, KendoUI

Expectations

* The code should be clean and understandable
* When possible the code should be separated into assemblies/modules
* The code’s critical parts (both backend and frontend) should be tested