**Assignment 2**

In this assignment you will implement a full stack application comprising:

* A web user interface (implemented in React, TypeScript, and CSS)
* A web server application (implemented in ASP.NET or Spring Boot)
* A database (SQL Server LocalDB or in-memory H2 database)

**Scenario**

DNB applications have a lot of configuration data that is spread around the organization. Your task is to implement a solution that consolidates the configuration data into a single database, with a web UI that enables users to view and manage the data.

**Data structure**

We suggest 3 tables in your database:

* **Registered administrators**

Some of the features in the application should only be available to administrators. Your application must be able to authenticate administrators, to permit them to access these features. To facilitate authentication, we suggest a simple table of registered administrators**.** Each registered administrator has a username and password. For simplicity in this assignment, you can insert a few hard-coded entries in this table.

* **Environments**

Configuration data in DNB is organized into *environments*. For example, there might be environments such as "development", "test", "production", "disaster recovery", and so on. You will need a table to hold a list of these environments. Each environment has a short name (e.g. "DEV") and a longer descriptive name (e.g. "development"). The list of environments isn't hard-coded - your application will allow administrators to edit and add environments.

* **Configuration data**

This is the main table in the database. Each item of configuration data has an environment, a key name, a value, and a timestamp indicating when it was last modified.

**Application functionality**

The application must offer a web UI that makes it easy for administrators to edit configuration data, and for any user to view configuration data. Implement a SPA with the following screens:

* **Welcome screen**

Displays summary info and some nice graphic.

* **View Configuration Data screen**

This screen allows any user to view configuration data. This screen should allow users to do the following:

* + List available environments.
  + For a specific environment, list the configuration data for that environment. There might be a lot of configuration data, so think about implementing paging.
* **Admin screen**

This screen allows administrators to manage environments and configuration data. This screen should work as follows…

First things first, ask the user to enter a user name and password, and submit to the server to authenticate as an administrator. The server should return an authentication token. All subsequent communication between the web UI and the server should pass this token as an HTTP header. The server should only allow administrative tasks to proceed if there's an accompanying authentication token in the request.

For authenticated administrators, offer the following features:

* + Edit the description for an existing environment.
  + Add a new environment.
  + Add, edit, or delete an item of configuration data.
  + View configuration data modifications within a specified timeframe.

**Note: Consult the "DNB Patterns and Best Practices" document and try to adopt these suggestions in your React code wherever possible.**

**Presentations**

On Monday 12 April you'll be invited to present your solution to the other teams. Aim for about 25 minutes, plus 5 minutes of Q&A. During your presentation:

* Run the application to demonstrate its functionality and user interface
* Explain your database structure
* Walk through key features in your code (both the client-side and server-side)
* Discuss your approach to testing
* Discuss limitations and possible extensions