Acceso a Datos

UT05 Spring Boot 2

Proyecto – API REST Segura 2

Prof. [Diego Linares Ortiz]

UT05 Proyecto
API REST Segura -2

Objetivo

En este ejercicio práctico vas a realizar la implementación de una API REST segura donde aplicaremos todas las técnicas que hemos visto hasta ahora. **Es imprescindible que vayas paso a paso, para así asegurar que la aplicación se construye sobre unos cimientos sólidos y estables.**

El objetivo del proyecto es que realices una API REST segura desde principio hasta el fin. Deberás terminar teniendo una API que cumplirá con los principios SOLID, segura, probada y bien documentada.

La API REST que se debe implementar será el back-end de una aplicación sencilla de gestión de tareas del hogar. Esta aplicación permitirá hacer login, registrarse en la aplicación, ver las tareas que un usuario tiene asignadas, marcarlas como hechas y eliminarlas si lo desea. La aplicación tendrá una parte de administrador que permitirá marcar como hecha o eliminar cualquier tarea de cualquier usuario. Ambos tipos de usuario tendrán a su disposición una funcionalidad para dar de alta tareas, el administrador podrá dar de alta tareas a cualquier usuario y otro usuario sólo se puede dar de alta tareas a sí mismo.

Resumen de la aplicación

Aquí se presenta un resumen de las funcionalidades que deberá tener la aplicación:

1. Todos, sin restricciones:
   - Login
   - Registro
2. Usuario con rol USER
   - Ver sus tareas
   - Marcar como hecha una tarea propia
   - Eliminar una tarea propia
   - Darse de alta una tarea
3. Usuario con rol Admin
   - Ver todas las tareas
   - Eliminar cualquier tarea de cualquier usuario
   - Dar de alta tareas a cualquier usuario

Enunciado

Construye e implementa una API REST segura siguiendo las siguientes restricciones:

1. Crea un repositorio en GitHub, que sea público y que tenga al profesor como colaborador directo del repositorio. Crea un README.md donde plantees los siguientes puntos:
    a. Nombre del proyecto
    b. Descripción detallada de los documentos que intervendrán en el proyecto, así como sus campos.
2. En el README anteriormente construido deberás incluir lo siguiente (aparte de lo ya descrito)
    a. Indicar los endpoints que se van a desarrollar para cada documento.
    b. Describir cada uno de los endpoints. Realiza una explicación sencilla de cada endpoint.
    c. Describe la lógica de negocio que va a contener tu aplicación.
    d. Describe las excepciones que vas a generar y los códigos de estado que vas a poner en todos los casos.
    e. Describe las restricciones de seguridad que vas a aplicar dentro de tu API REST

    SI FALTARA CUALQUIER PUNTO DE LOS ANTERIORMENTE DESCRITOS, EL PROYECTO SE DARÁ COMO INVÁLIDO
3. Realiza la implementación de la API siguiendo las directrices que has marcado en la parte de la documentación que ya has desarrollado. La aplicación deberá cumplir con los siguientes mínimos:
    a. Debe haber una entidad **Usuario** que:
        i. Tenga al menos los campos ***username, password y roles***.
        ii. La password debe almacenarse "*hasheada*" en la BDD
    b. Habrá 2 entidades más, una llamada **Direccion** y otra llamada **Tarea**:
    c. Debes implementar la seguridad de la aplicación haciendo uso de Spring Security y que cumpla:
        i. Que se usa un cifrado asimétrico por clave pública y privada para el control de acceso.
        ii. Que se usa JWT para el control de acceso.
        iii. Que hay variedad de restricciones de control de acceso a los endpoints (*lo mismo que el punto b.ii*)
4. Una vez realizada la implementación de la API, realiza pruebas del buen funcionamiento de la misma.
    a. Usa insomnia, swagger o postman para probar los diferentes endpoints de la API
    b. Plantea las pruebas que vas a realizar para comprobar el buen funcionamiento de la API implementada
    c. Documenta las pruebas realizadas para mostrar que la API ha cumplido con su funcionamiento.

5. Cuando finalices toda la implementación y las pruebas podrás concluir el proyecto desplegando la API en una plataforma *cloud* gratuita como Render.
    a. Cuando despliegues la aplicación, realiza pruebas de funcionamiento haciendo uso de la interfaz gráfica que muestren que la aplicación ha sido correctamente desplegada y funciona correctamente. **Que funcione correctamente significa que se comporta bien en todos los casos, tanto los correctos como los erróneos**
    b. Realiza un vídeo donde muestres estas pruebas y comentes sobre las mismas.

Conclusión y entrega

Para concluir el proyecto deberás tener lo siguiente:

1. Repositorio de Github público con todo el código de la API desarrollada.
2. README.md que incluye:
    a. Todos los puntos teóricos indicados en los puntos 1 y 2 del enunciado
    b. Todas las pruebas realizadas a la API REST haciendo uso de Insomnia (o Postman o Swagger).
    c. Un vídeo (o pantallazos) con las pruebas realizadas una vez desplegada la aplicación haciendo uso de la interfaz gráfica.
3. Un despliegue en Render de la API.

RAs y CEs evaluados

5. Desarrolla aplicaciones que gestionan la información almacenada en bases de datos nativas XML evaluando y utilizando clases específicas.

a) Se han valorado las ventajas e inconvenientes de utilizar una base de datos nativa XML.

b) Se ha instalado el gestor de base de datos.

c) Se ha configurado el gestor de base de datos.

d) Se ha establecido la conexión con la base de datos.

e) Se han desarrollado aplicaciones que efectúan consultas sobre el contenido de la base de datos.

f) Se han añadido y eliminado colecciones de la base de datos.

g) Se han desarrollado aplicaciones para añadir, modificar y eliminar documentos XML de la base de datos.

6. Programa componentes de acceso a datos identificando las características que debe poseer un componente y utilizando herramientas de desarrollo.

a) Se han valorado las ventajas e inconvenientes de utilizar programación orientada a componentes.

b) Se han identificado herramientas de desarrollo de componentes.

c) Se han programado componentes que gestionan información almacenada en ficheros.

d) Se han programado componentes que gestionan mediante conectores información almacenada en bases de datos.

e) Se han programado componentes que gestionan información usando mapeo objeto relacional.

f) Se han programado componentes que gestionan información almacenada en bases de datos objeto relacionales y orientadas a objetos.

g) Se han programado componentes que gestionan información almacenada en una base de datos nativa XML.

h) Se han probado y documentado los componentes desarrollados.

i) Se han integrado los componentes desarrollados en aplicaciones.

Indicadores	Niveles de logro		
	Bien	Regular	Insuficiente

	100%	50%	0%