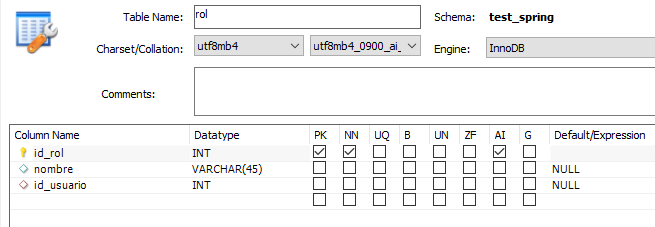
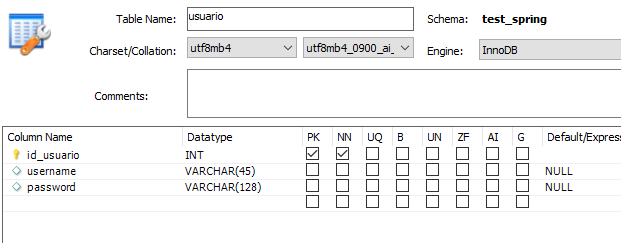
121. Seguridad con Spring y Hibernate/JPA

Pt1

Se modificara la app control cliente para manejar el concetp de seguridada usando una DB y JPA

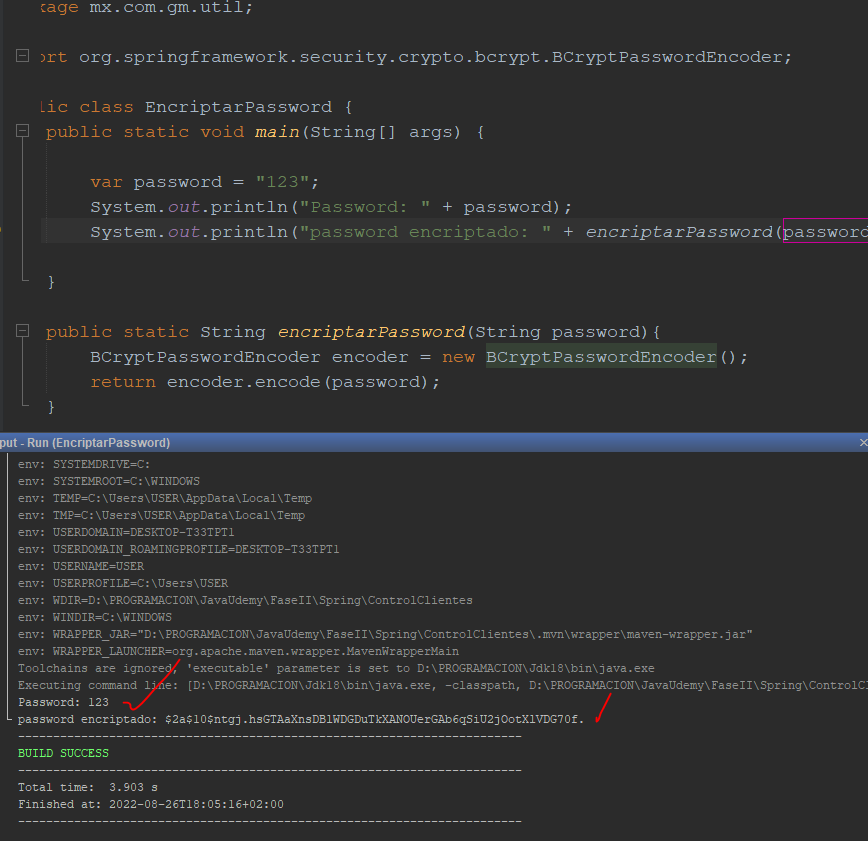
1. Crear tablas usuario y roll (con el fin de no tener en codigo duro el password y los roles si no que vengan de una DB)



Pt2

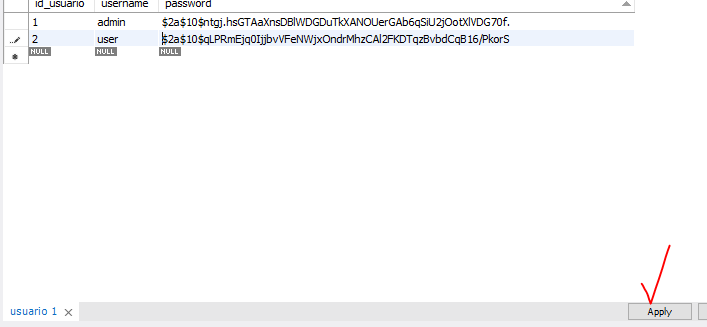
Crear clase de utileria para poder generar los password que vamos a utilizar en nuetra DB, se deben generar de manera separa a menos que se haga todo el flujo para crear un nuevo usuario pero se realizaran por separa para simplificiar

1. Crear clase main EncriptarPassword (la idea e tener un código encriptado para la clave se puede ejecutar cuantas veces quiera y obtener un nuevo codigo)



Como se aprecia esta clase lo que hace es generar el password encriptado a utilizar, todo esto se puede hacer desde la app pero se debe hacer todo el flujo para crear nuevos usuarios y agregar nuevos password y utilice ese tipo de encripcion, estaqmos enfocados en remplazar la configuración de seguridad creada y de ese modo utilizar los usuario de la DB y no en código duro.

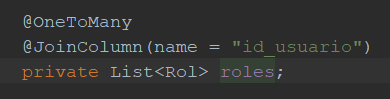
1. El código encriptado generado se agrega en la DB



Pt3

Mapeo de la clases usuario y Rol

1. Crear clase domain.Rol (Entity)
   1. El mapeo de IdUsuario se realiza desde el mapeo de la tabla que es su idPrincipal en este caso de la tabla usuario
   2. Mapeo un usuario puede tener múltiples roles ---- OneToMany



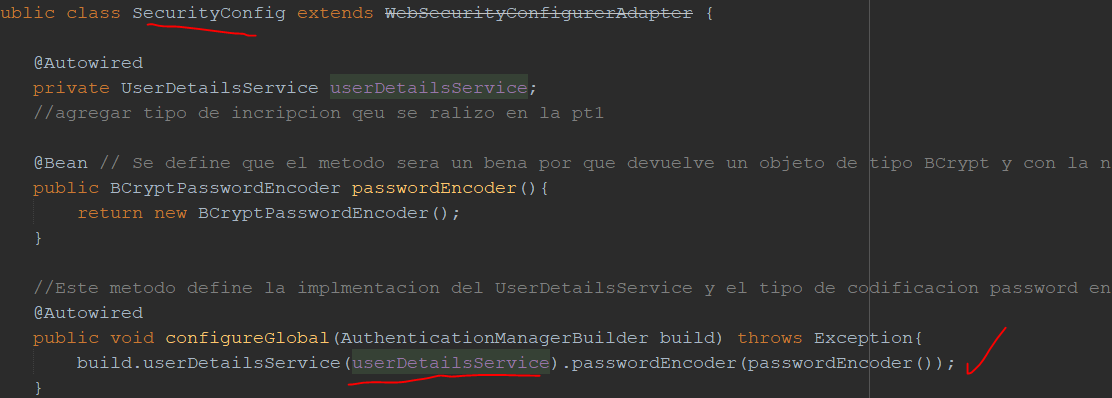
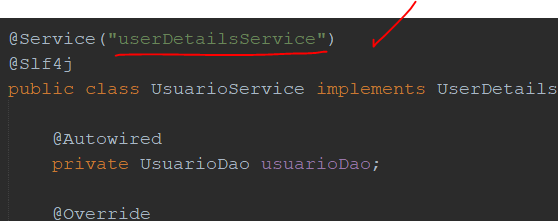
Para no definir toda la funcionalidad de seguridad spring facilita con clases creadas para permitir fácilmente la configuración

1. Crear interface UsuarioDAO
   1. Extiende de JpaRepository (Se puede usar CrudRepositorio aunque JpaRepositoriy es más completo)quien contiene todos los métodos CRUD mas comunes
   2. Aplicaremos la Seguridad de spring creando el método finByUsername(String username)
2. Definir servicio de la interface UsuarioDao
   1. Convertir clase en clase de servicio para que pueda ser usada por Spring Security
      1. Agregar anotación @Service("userDetailsService")
   2. Esta clase lo que permite es omitir toda la clase de SecurityConfig que es mapear todos los roles en memoria de este modo será mas practico y dinámico desde la DB

Pt4

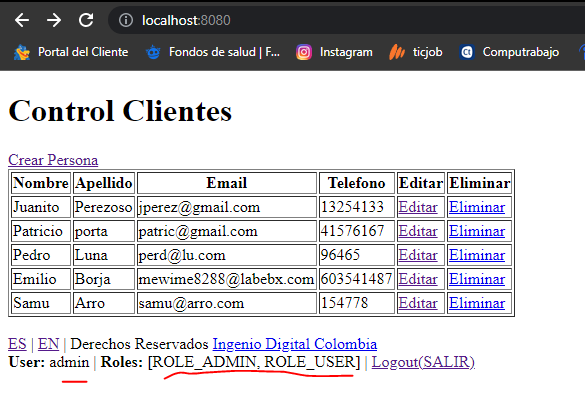
Cargar la implementación de Jpa omitiendo en disco duro la clase Security config.

1. Modificar clase SecurityConfig
2. Injectar la clase UserDetailservice de spring
3. Mapear el tipo de objeto ENCRIPTADO que se enviara al BEAN
4. Injectar metodo define la implementación del UserDetailsService y el tipo de codificacion password encoder. Recordar el que nombre de la instancia de sevicio se llama userDetailservice

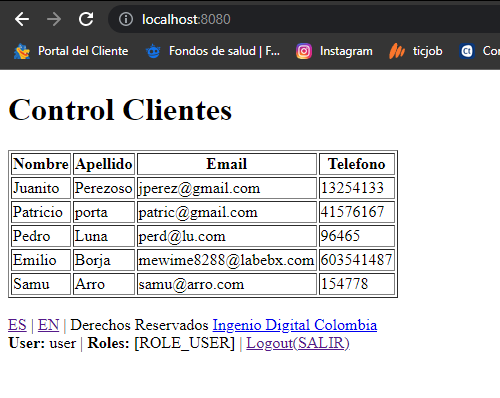


* Prueba General

Para admin



Para user



PREGUNTA SIN RESOLVER : Con la anotación Bean no funciona el método de encriptar pero sin anotación funciona correctamente la app ¿Por qué pasa esto?

