

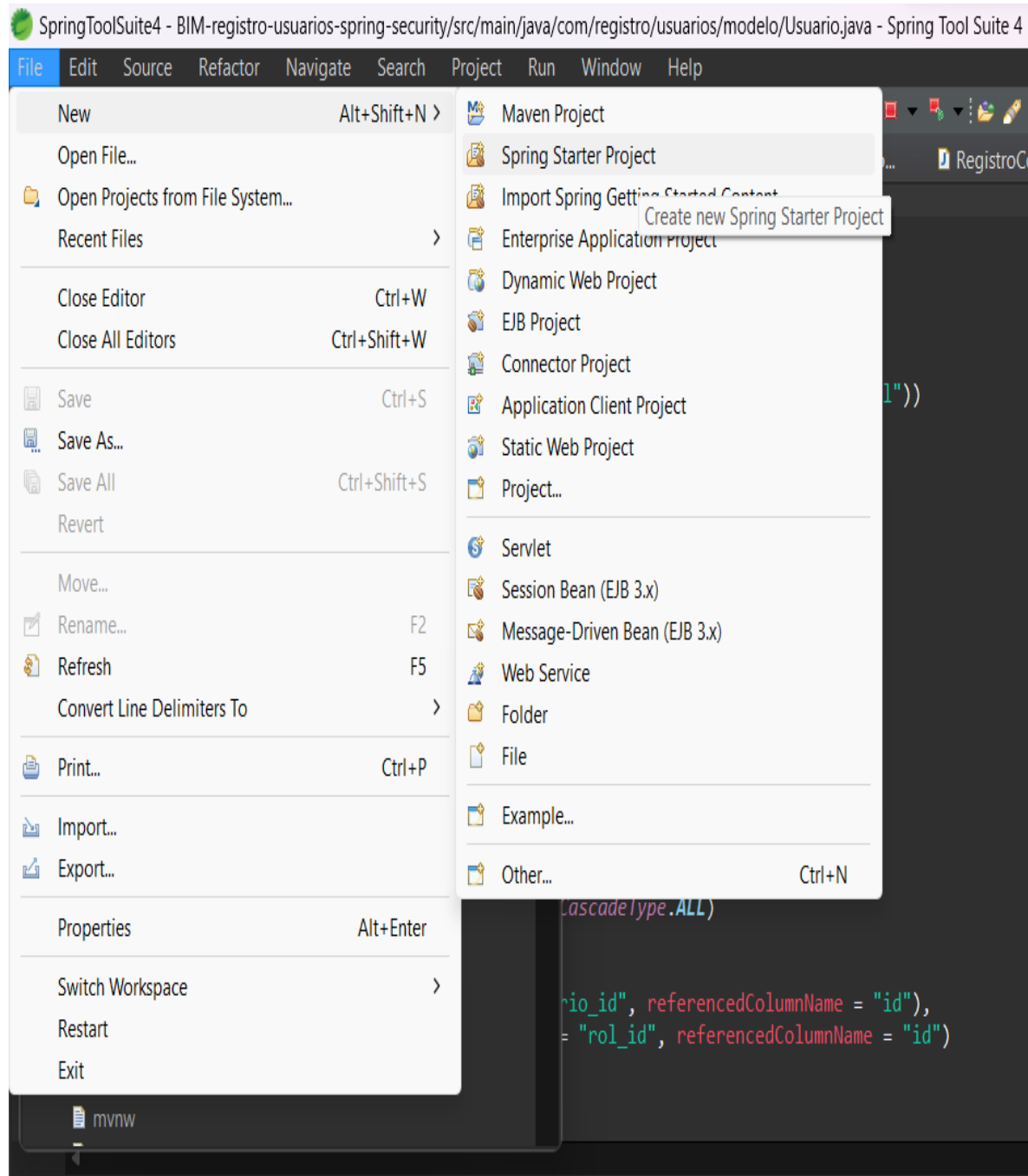
BIM
(BANCO INMOVILIARIO
MEXICANO)

PRUEBA TÉCNICA PARA
DESARROLLADOR JAVA

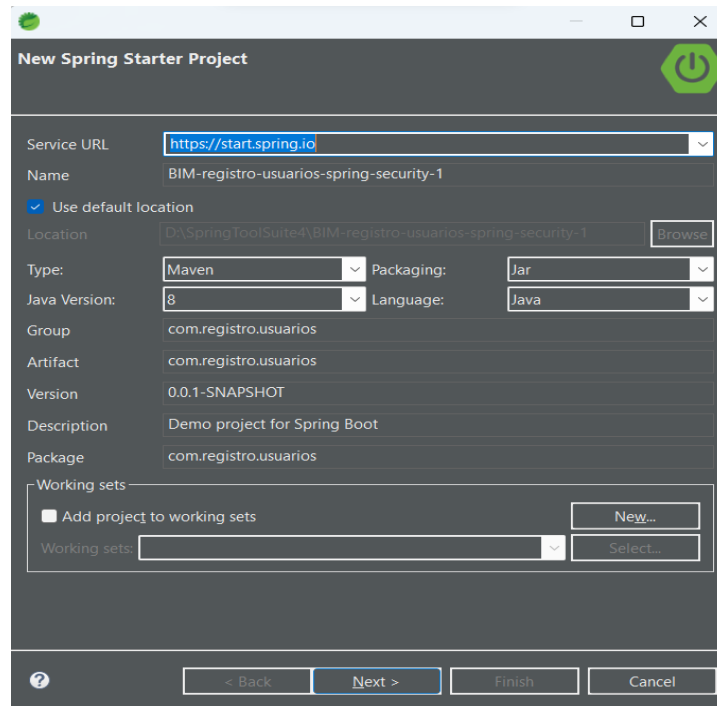
ING. SEBASTIÁN GERARDO
PALACIOS PÉREZ

CREACIÓN DEL PROYECTO

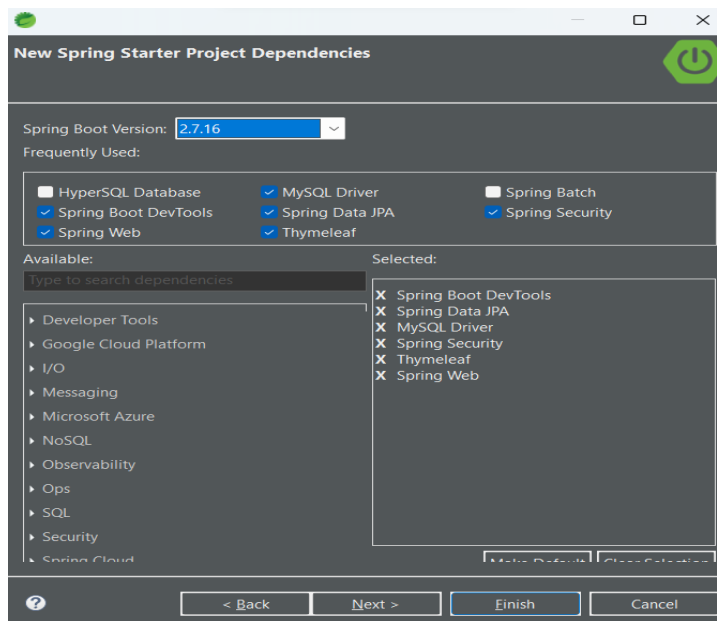
Iniciando con la creación del proyecto utilizando Spring Tool Suite 4, creando un proyecto Spring de inicio (Spring Starter Project).



Colocando el nombre al group, artifact y package (usualmente por defecto se coloca el mismo que se ponga en el de group), utilice la versión 0.0.1 SNAPSHOT, lo realicé de tipo Maven con paquetería Jar, la versión de Java utilicé la 8 pues es la más usada en el ámbito por su accesibilidad.



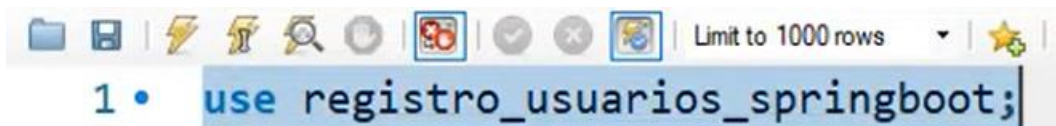
La versión más acoplada al proyecto que consideré de Spring boot fue la 2.7.16, ya que hay errores con las nuevas versiones, agregué las dependencias que utilicé en el proyecto, las cuales son Spring Boot DevTools, Spring Data JPA, MySQL Driver, Spring Security, Thymeleaf y Spring Web



Mientras se descargan los componentes necesarios del proyecto y se va creando a la vez, utilizando MySQL Workbench creo la base de datos que utilizaré para dicho proyecto.

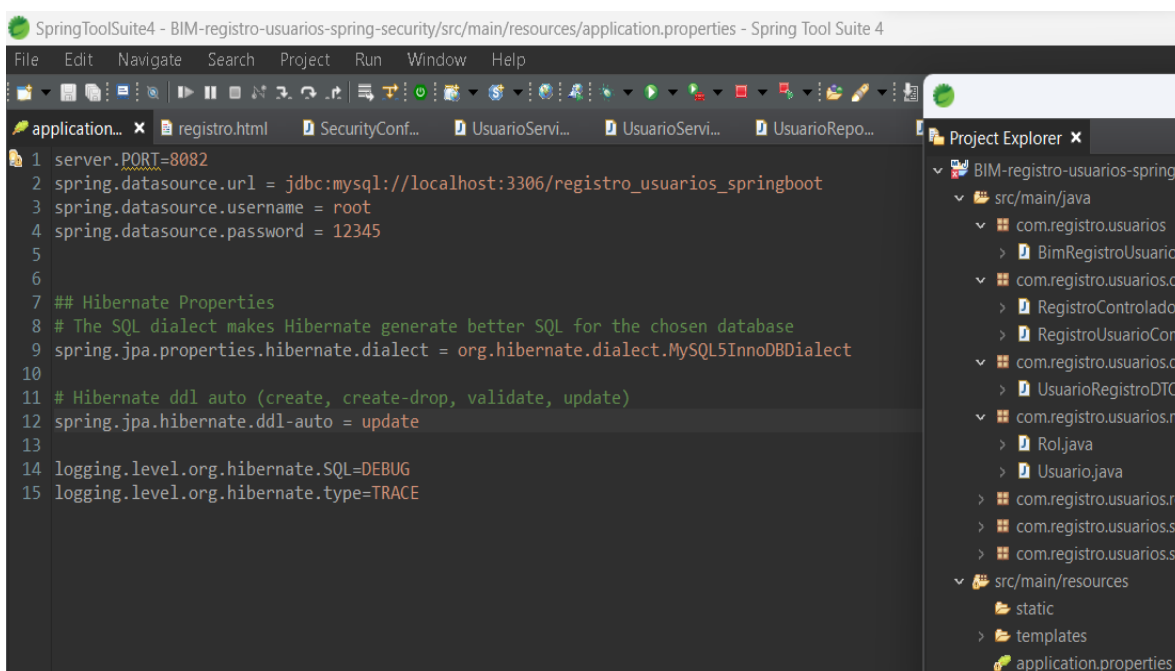


Con el comando anterior al haberlo creado, se coloca el comando para utilizarla (use) y simplemente dejarla así, pues las modificaciones y declaración de llaves y tipo de datos que llevara la base, se irán programando en Spring Tool.



Al haberse terminado de crear y descargar los componentes del proyecto, ingresé en la ruta dando clic al proyecto, src, main, resources y desplegará la parte de “application.properties” dónde se le asignará un puerto para poder visualizar la función en la web (la dependencia de Spring Web), se le debe colocar la ruta con el nombre de la base de datos creada anteriormente

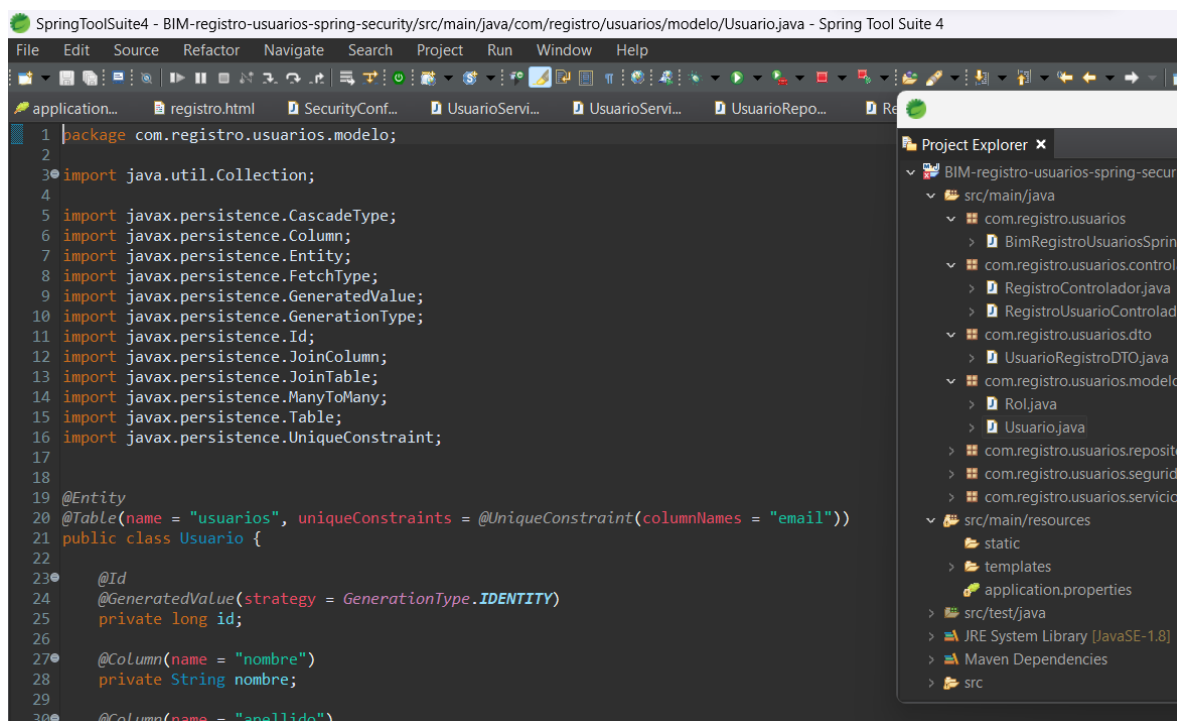
“registro_usuarios_springboot” colocando la contraseña y el nombre de usuario que se utiliza para ingresar a la misma, así se automatiza toda la modificación que se le realice al proyecto, colocando también el dialecto y usando hibernate en update, con el fin de que se vayan agregando datos sin perder el anterior que se haya agregado.



INICIO DE CODIGO

Sobre el paquete creado al hacer el proyecto (com.registro.usuarios) cree el nuevo paquete llamado "com.registro.usuarios.modelo" pues será el modelo para iniciar el proyecto, dentro del mencionado paquete vacío, se realiza la clase "Usuario", la cual llevará los componentes iniciales.

Agregué el comando para hacer la entidad, mapeada con una tabla que hice desde el editor llamada "usuarios", esa tabla que se va a crear debe llevar el uniqueConstraints para llevar el nombre de la columna único llamado email, declarando cada una de las propiedades llevarán los campos y su tipo de dato, así como la colección de roles, no sin antes, crear el ID como único y dando la estrategia con GeneratedValue y especificando IDENTITY.

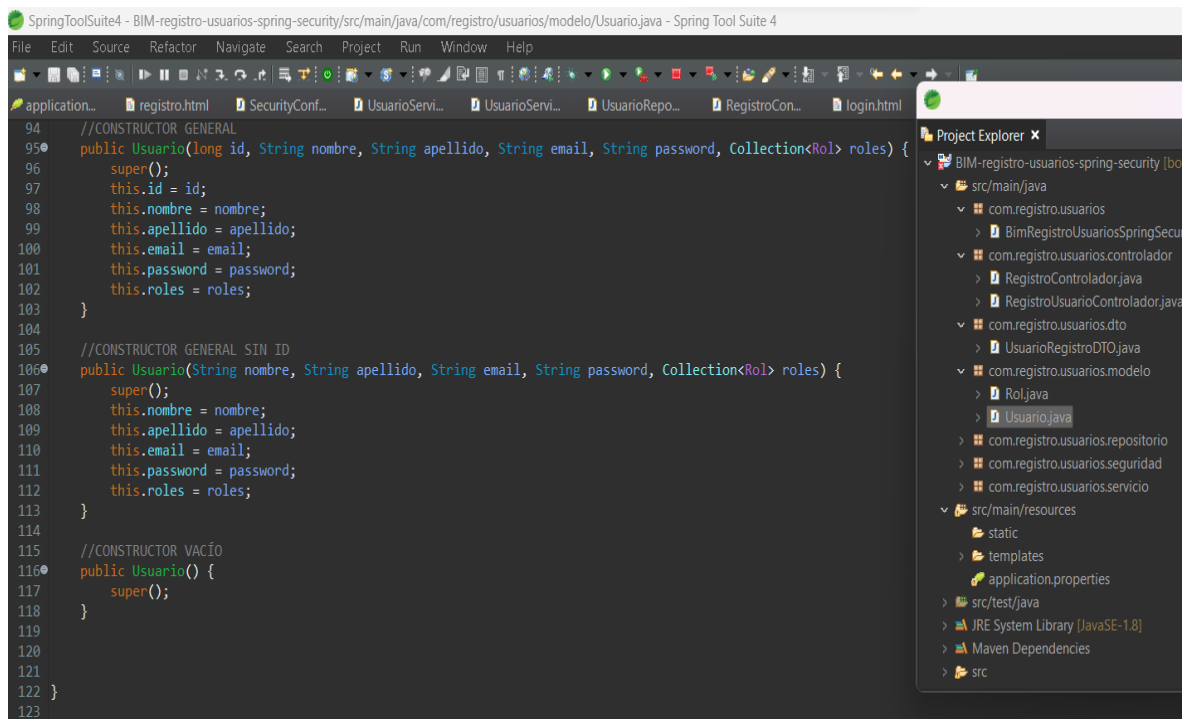
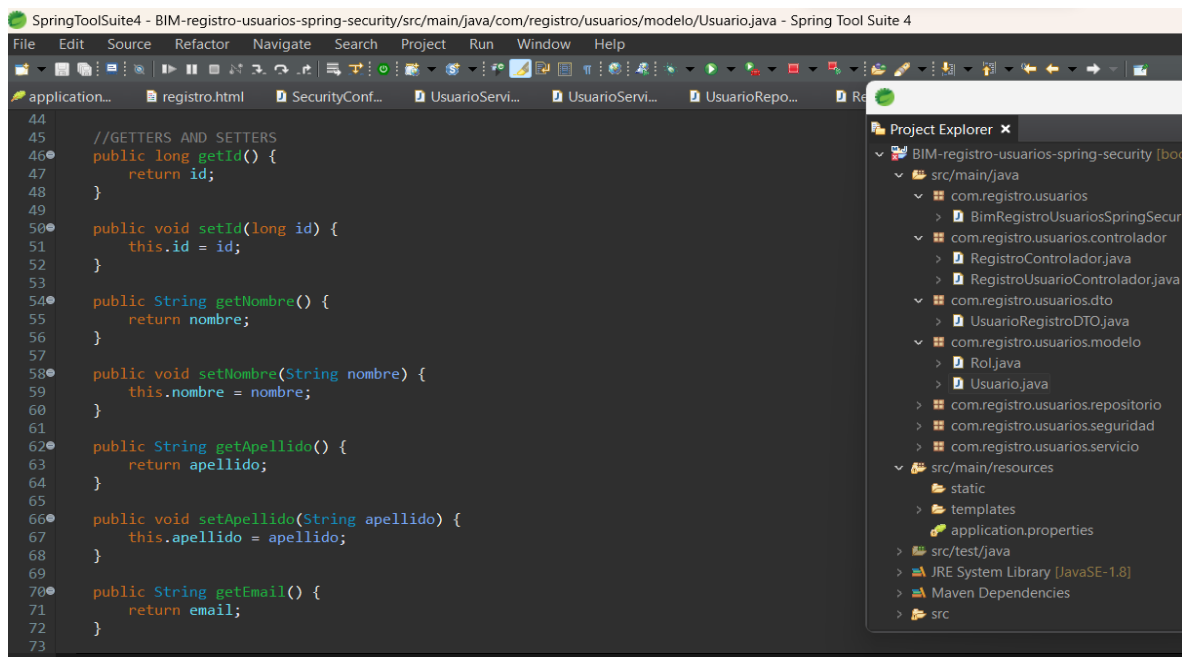


```
SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/java/com/registro/usuarios/modelo/Usuario.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
application... registro.html SecurityConf... UsuarioServi... UsuarioServi... UsuarioRepo...
1 package com.registro.usuarios.modelo;
2
3 import java.util.Collection;
4
5 import javax.persistence.CascadeType;
6 import javax.persistence.Column;
7 import javax.persistence.Entity;
8 import javax.persistence.FetchType;
9 import javax.persistence.GeneratedValue;
10 import javax.persistence.GenerationType;
11 import javax.persistence.Id;
12 import javax.persistence.JoinColumn;
13 import javax.persistence.JoinTable;
14 import javax.persistence.ManyToMany;
15 import javax.persistence.Table;
16 import javax.persistence.UniqueConstraint;
17
18
19 @Entity
20 @Table(name = "usuarios", uniqueConstraints = @UniqueConstraint(columnNames = "email"))
21 public class Usuario {
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     private long id;
26
27     @Column(name = "nombre")
28     private String nombre;
29
30     @Column(name = "apellido")
```

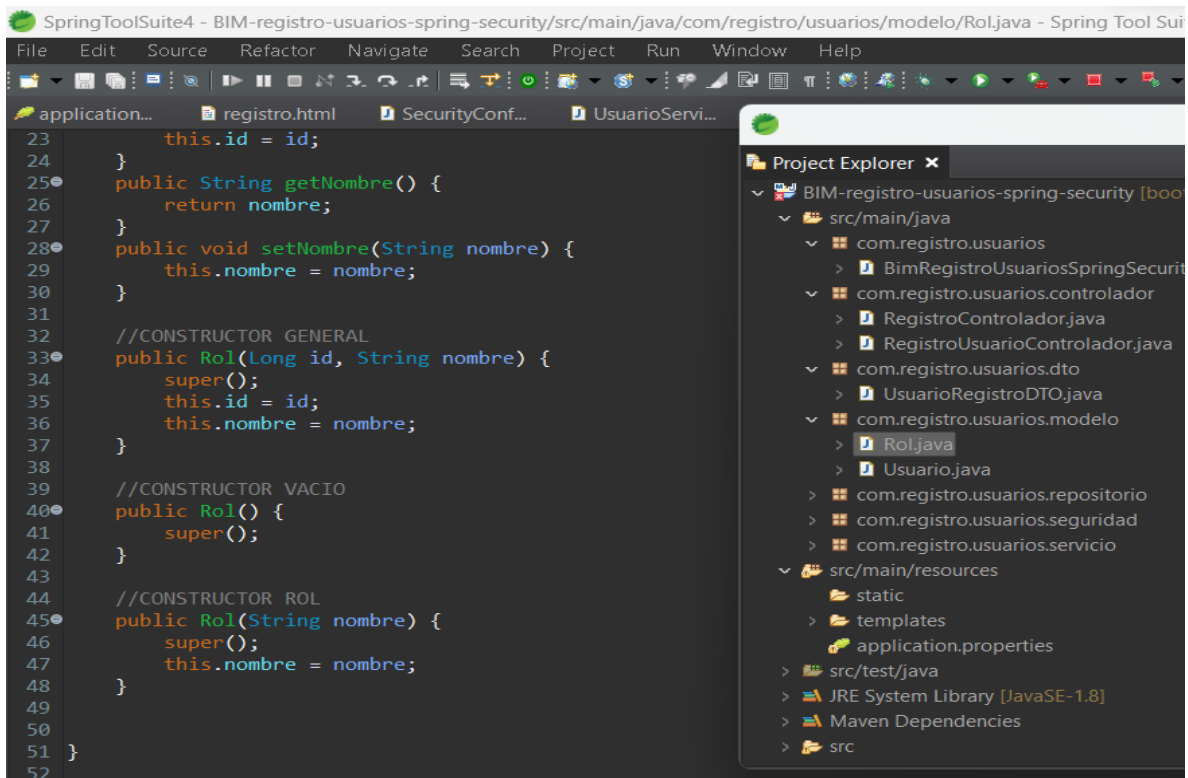
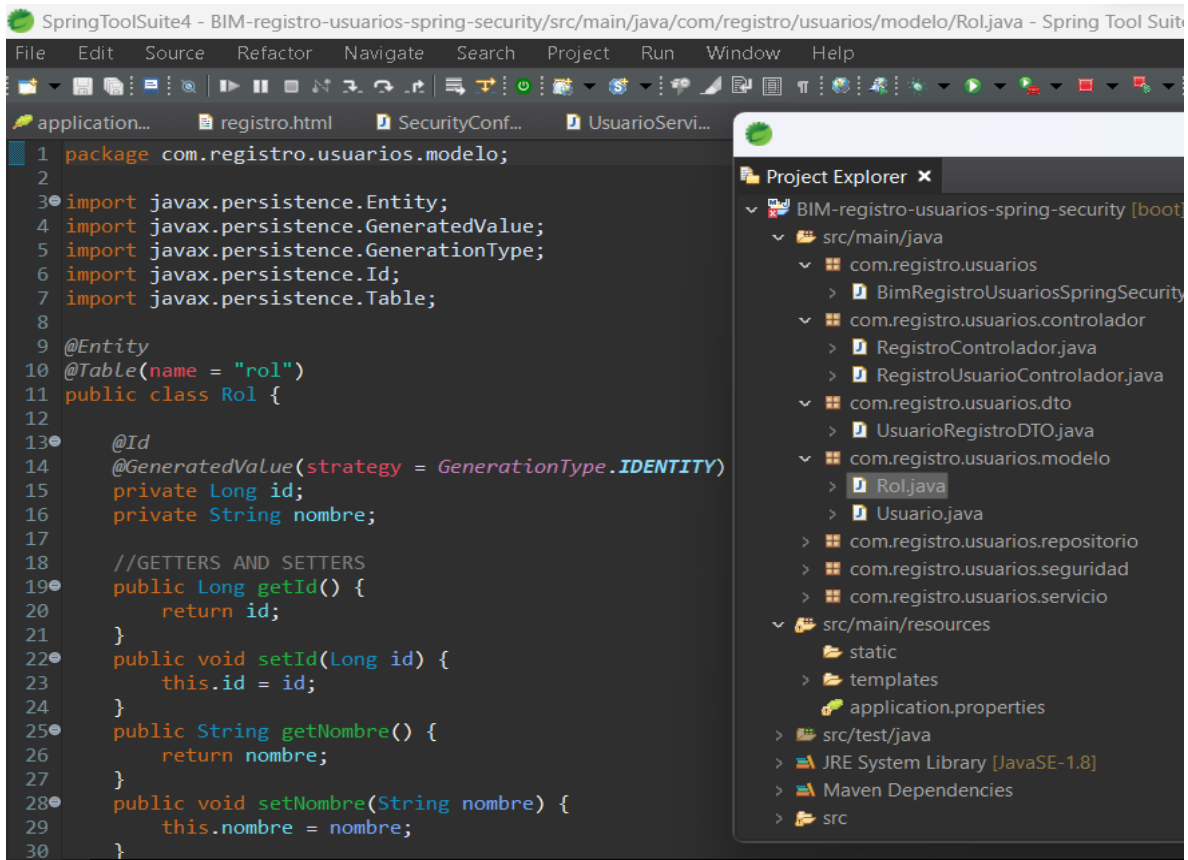
Project Explorer

- BIM-registro-usuarios-spring-security
 - src/main/java
 - com.registro.usuarios
 - BimRegistroUsuariosSpring
 - com.registro.usuarios.controla
 - RegistroControlador.java
 - RegistroUsuarioControlado
 - com.registro.usuarios.dto
 - UsuarioRegistroDTO.java
 - com.registro.usuarios.modelo
 - Rol.java
 - Usuario.java
 - com.registro.usuarios.reposito
 - com.registro.usuarios.seguridad
 - com.registro.usuarios.servicio
 - src/main/resources
 - static
 - templates
 - application.properties
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src

Se hace la generación de getters and setters en source, además de los constructores, uno sin el ID, otro completo y uno vacío para la captación de errores.



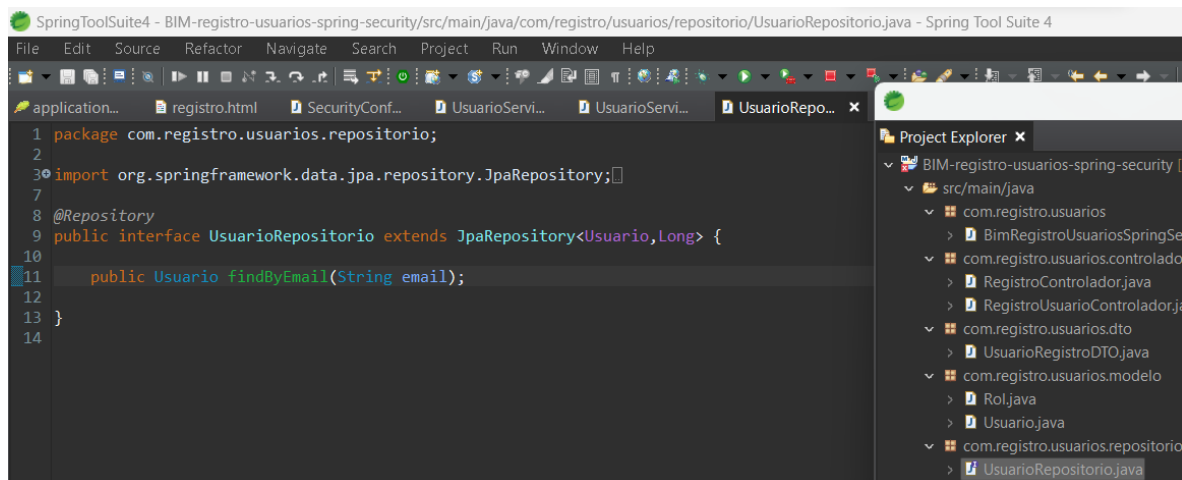
Para hacer funcional la colección de roles, se debe crear la clase rol, dónde se colocan los atributos correspondientes, tal como la entidad y la estrategia usada en GeneratedValue, así como los Getters and setters, también los constructores generales, el vacío y obviamente el de rol.



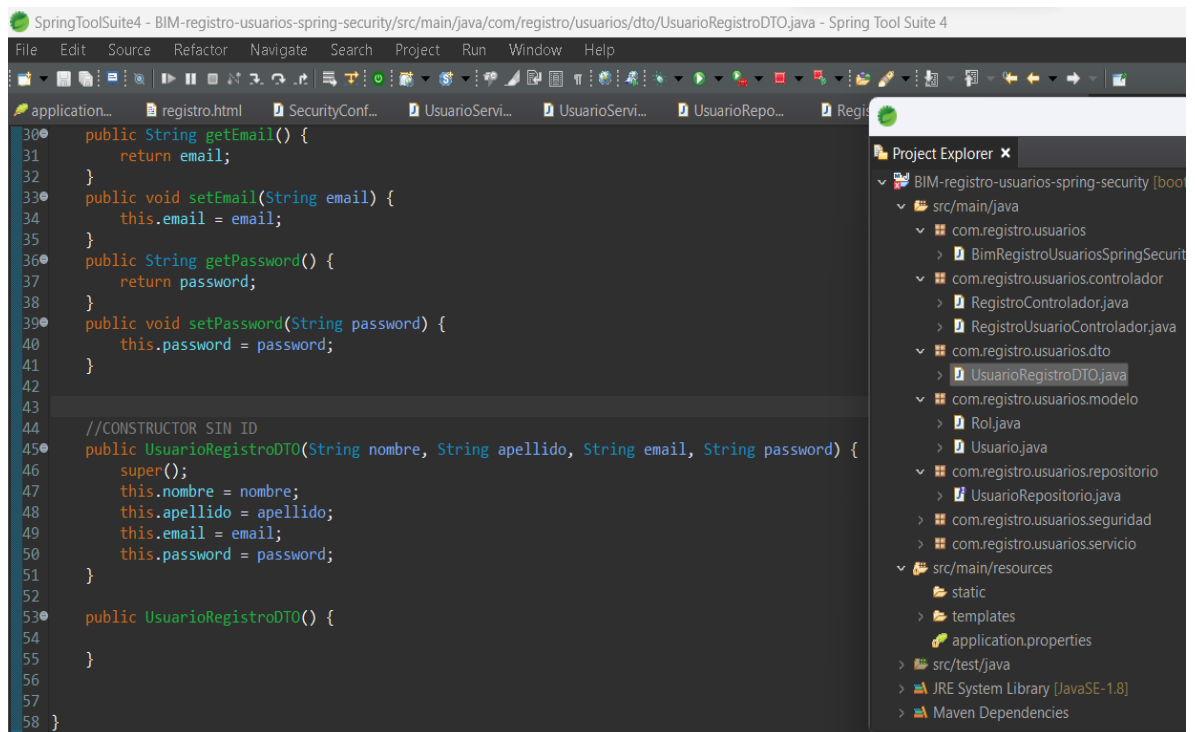
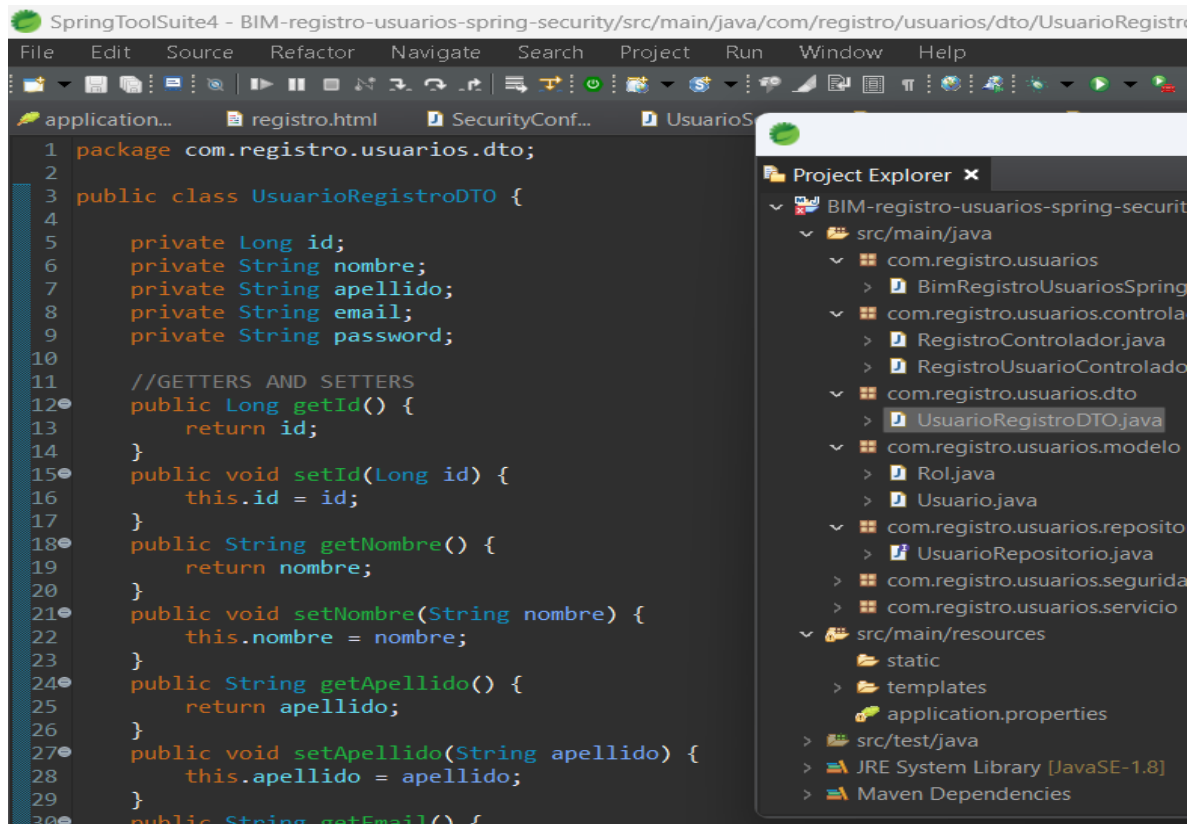
En la clase usuario, se debe realizar la relación correspondiente, se debe hacer la relación muchos a muchos "many to many" debido a que un usuario puede tener muchos roles, así como los roles, pueden ser hechos por muchos usuarios, el cual se debe crear una tabla intermediaria se debe normalizar dicha acción, esto hecho entre la declaración de atributos y los getters and setters.

```
35
36 @ManyToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)
37 @JoinTable(
38     name = "usuarios_roles",
39     joinColumns = @JoinColumn(name = "usuario_id", referencedColumnName = "id"),
40     inverseJoinColumns = @JoinColumn(name = "rol_id", referencedColumnName = "id")
41 )
42
43 private Collection<Rol>roles;
44
```

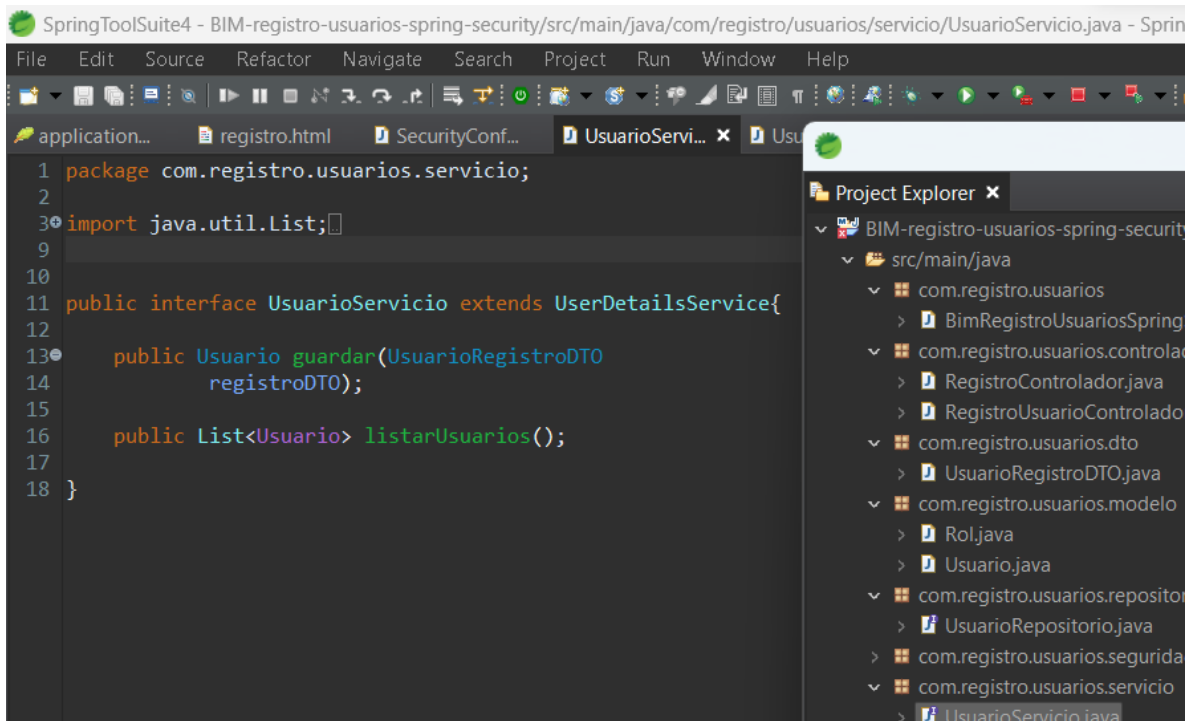
se debe crear el paquete de com.registro.usuarios.repositorio para hacer el repositorio (valga la redundancia) dentro del primer paquete creado el cual será raíz de todos los demás (com.registro.usuarios) la clase UsuarioRepositorio creada dentro del paquete recién hecho, haciendo la herencia de JpaRepository indicando la clase, así se implementan todos los metodos sin reescribir código.



Cree un nuevo paquete sobre el principal (com.registro.usuarios) llamado "com.registro.usuarios.dto" al cual se le crea una nueva clase "UsuarioRegistroDTO", haciendo básicamente los mismos atributos de la clase Usuario, generando sus respectivos getters and setters y además el constructor sin ID y por supuesto, el constructor vacío.



Ahora pasé a generar el servicio, sobre el paquete principal, se crea un nuevo paquete vacío llamado "com.registro.usuarios.servicio", al cual se le hace una nueva interfaz llamada "UsuarioServicio" el cual sólo tendrá los métodos.

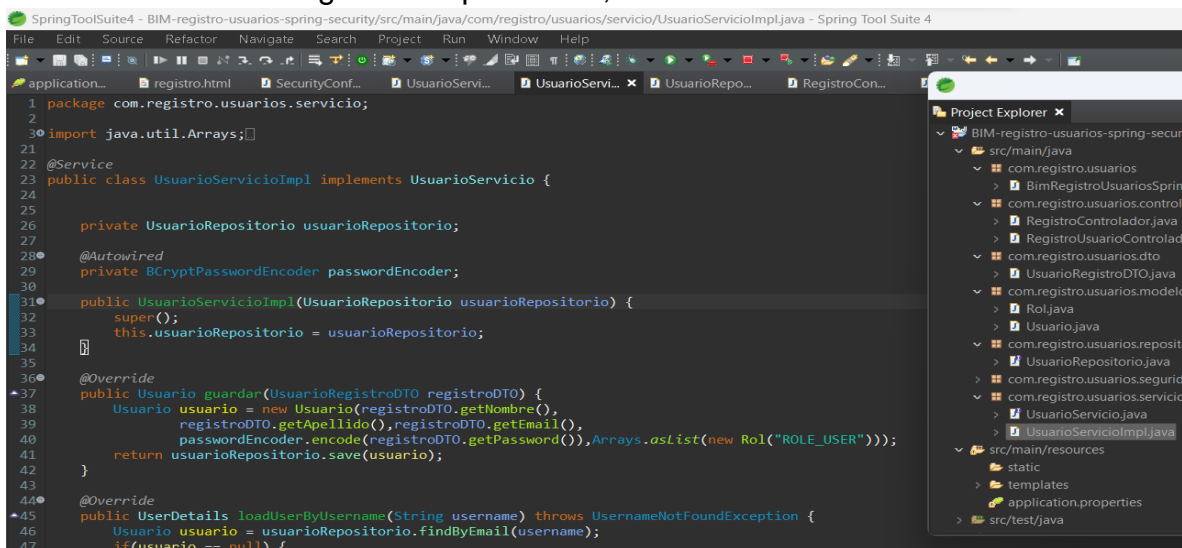


The screenshot shows the SpringToolSuite4 IDE with the file `UsuarioServicio.java` open. The code defines a package `com.registro.usuarios.servicio` and imports `java.util.List`. It then defines a public interface `UsuarioServicio` that extends `UserDetailsService`. The interface contains two methods: `guardar(UsuarioRegistroDTO registroDTO)` and `listarUsuarios()`.

```
1 package com.registro.usuarios.servicio;
2
3 import java.util.List;
4
5
6
7
8
9
10
11 public interface UsuarioServicio extends UserDetailsService{
12
13     public Usuario guardar(UsuarioRegistroDTO
14         registroDTO);
15
16     public List<Usuario> listarUsuarios();
17 }
18
```

The Project Explorer on the right shows the project structure, with the `UsuarioServicio.java` file highlighted under the `com.registro.usuarios.servicio` package.

Ahora se debe hacer la implementación de la interfaz, a lo cual se tiene que crear una nueva clase dentro de este mismo paquete (`com.registro.usuarios.servicio`) llamada "UsuarioServicioImpl", la cual va a implementar el método de guardar además del UsuarioServicio de la interfaz hecha anteriormente, llamando el repositorio, originando constructor del mismo, en el método guardar se debe establecer completamente para todos los parámetros declarados y puedan ser almacenados en su lugar correspondiente, así como también los rol establecidos.



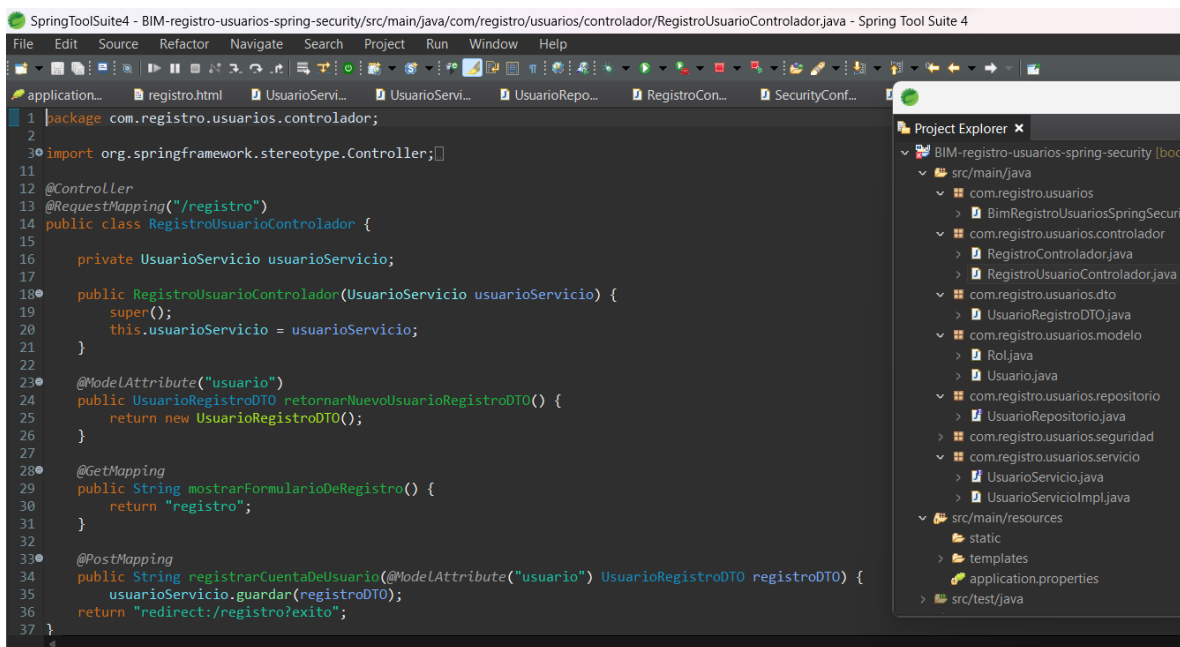
The screenshot shows the SpringToolSuite4 IDE with the file `UsuarioServicioImpl.java` open. The code defines a package `com.registro.usuarios.servicio` and imports `java.util.Arrays`. It then defines a public class `UsuarioServicioImpl` that implements `UsuarioServicio`. The class has two private fields: `UsuarioRepositorio usuarioRepositorio` and `BCryptPasswordEncoder passwordEncoder`. It has a constructor that takes a `UsuarioRepositorio` and a `BCryptPasswordEncoder` as arguments. It also has two methods: `guardar(UsuarioRegistroDTO registroDTO)` and `loadUserByUsername(String username)`.

```
1 package com.registro.usuarios.servicio;
2
3 import java.util.Arrays;
4
5
6
7
8
9
10
11 @Service
12 public class UsuarioServicioImpl implements UsuarioServicio {
13
14     private UsuarioRepositorio usuarioRepositorio;
15
16     @Autowired
17     private BCryptPasswordEncoder passwordEncoder;
18
19     public UsuarioServicioImpl(UsuarioRepositorio usuarioRepositorio) {
20         super();
21         this.usuarioRepositorio = usuarioRepositorio;
22     }
23
24     @Override
25     public Usuario guardar(UsuarioRegistroDTO registroDTO) {
26         Usuario usuario = new Usuario(registroDTO.getNombre(),
27             registroDTO.getApellido(), registroDTO.getEmail(),
28             passwordEncoder.encode(registroDTO.getPassword()), Arrays.asList(new Rol("ROLE_USER")));
29         return usuarioRepositorio.save(usuario);
30     }
31
32     @Override
33     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
34         Usuario usuario = usuarioRepositorio.findByEmail(username);
35         if(usuario == null) {
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

The Project Explorer on the right shows the project structure, with the `UsuarioServicioImpl.java` file highlighted under the `com.registro.usuarios.servicio` package.

Continuando sobre el paquete principal, se crea uno nuevo agregando la extensión "controlador", haciendo también una nueva clase sobre este paquete vacío llamado "RegistroUsuarioControlador", básicamente lo que se le debe añadir es la anotación de controller y el requestMapping, indicando que si quiero acceder al controlador, generando su constructor correspondiente, básicamente el funcionamiento de este controlador es para controlar las URLs, también colocarle la función de retornar a un archivo html cuando se necesite una petición get, además de re direccionar al registro dando el mensaje de éxito.



PRIMER ARCHIVO HTML (REGISTRO)

Sobre src, main, resources, template... se crea un archivo html file, el cual se llamará registro.html, dónde tendrá el bootstrap tomado de la página, el elegido por mí, fue el 3.3.7 que es el que me agradó para esta prueba, además del título y componentes básicos para el front en cuanto al registro de usuarios, colocando también la ruta de thymeleaf pues se usan sus anotaciones y se debe referenciar, además de todos los mensajes que se pueden agregar al tener éxito en registro o viceversa pues se debe advertir cuando algo salió mal, así como los campos para registrar el nombre, apellido, email y password que serán almacenados en la base de datos, también agregando botones para regresar en caso de ya ser alguien registrado o bien simplemente volver a la página por defecto (el inicio de sesión). No sin colocar la contraseña con un type tipo password, pues no queremos nadie la sepa.

```

SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/resources/templates/registro.html - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
application... registro.html x UsuarioServi... UsuarioServi... UsuarioRepo... RegistroCon...

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <meta charset="utf-8">
5 <title>Registro de usuarios</title>
6 <!-- Latest compiled and minified CSS -->
7 <link rel="stylesheet"
8 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
9 integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
10 crossorigin="anonymous">
11
12 <!-- Optional theme -->
13 <link rel="stylesheet"
14 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"
15 integrity="sha384-rHyoN1iRsVX4nD0JutLnGasLJC7uC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwL/Sp"
16 crossorigin="anonymous">
17 </head>
18 <body>
19
20
21 <nav class="navbar navbar-inverse navbar-fixed-top">
22 <div class="container">
23 <div class="navbar-header">
24 <button type="button" class="navbar-toggle collapsed"
25 data-toggle="collapse" data-target="#navbar" aria-expanded="false"
26 aria-controls="navbar">
27 <span class="sr-only">Toggle navigation</span> <span
28 class="icon-bar"></span> <span class="icon-bar"></span> <span
29 class="icon-bar"></span>
30 </button>

```

```

SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/resources/templates/registro.html - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
application... registro.html x UsuarioServi... UsuarioServi... UsuarioRepo... RegistroCon... login...

31 </button>
32 <a class="navbar-brand" href="#" th:href="@{/}">Registro e
33 inicio de sesión</a>
34 </div>
35 </div>
36 </nav>
37
38 <br>
39
40 <div class="container">
41 <div class="row">
42 <div class="col-md-6 col-md-offset-3">
43 <div th:if="${param.exito}">
44 <div class="alert alert-info">Se ha registrado exitosamente a
45 la aplicación</div>
46 </div>
47
48 <h1>Registrate</h1>
49 <form th:action="@{/registro}" method="post" th:object="${usuario}">
50 <div class="form-group">
51 <label class="control-label" for="nombre">Nombre : </label> <input
52 id="nombre" type="text" class="form-control" th:field="*{nombre}"
53 required autofocus="autofocus">
54 </div>
55
56 <div class="form-group">
57 <label class="control-label" for="apellido">Apellido : </label> <input
58 id="apellido" class="form-control" th:field="*{apellido}">

```

```

84
85     <!-- Latest compiled and minified JavaScript -->
86     <script
87         src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
88         integrity="sha384-Tc5IQib027qvyjSMfHj0MaLkfuWVxZxUPnCJA7L2mCWNIpG9mGCD8wGNIcPD7Txa"
89         crossorigin="anonymous"></script>
90 </body>
91 </html>

```

SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/resources/templates/registro.html - Spring Tool Suite 4

File Edit Navigate Search Project Run Window Help

application... registro.html x UsuarioServ... UsuarioRepo... RegistroCon... login.html

```

56 <div class="form-group">
57     <label class="control-label" for="apellido">Apellido : </label> <input
58         id="apellido" class="form-control" th:field="*{apellido}"
59         required autofocus="autofocus">
60 </div>
61
62 <div class="form-group">
63     <label class="control-label" for="email">Email : </label> <input
64         id="email" type="email" class="form-control" th:field="*{email}"
65         required autofocus="autofocus">
66 </div>
67
68 <div class="form-group">
69     <label class="control-label" for="password">Password : </label> <input
70         type="password" id="password" class="form-control" th:field="*{password}"
71         required autofocus="autofocus">
72 </div>
73
74 <div class="form-group">
75     <button type="submit" class="btn btn-success">Registrar</button>
76     <span>Si estas registrado <a th:href="@{/login}">inicia sesión aquí</a></span>
77 </div>
78 </form>
79
80 </div>
81 </div>
82 </div>
83
84

```

La visualización del primer html que es el de registro, debe contener toda la información posible para una cómoda vista al usuario, ingresando al puerto establecido en application (8082) en localhost y visualizando que la contraseña sea oculta y nos indique si fue registrado exitosamente la persona.

Registro e inicio de sesión

Se ha registrado exitosamente a la aplicación

Regístrate

Nombre :

Gerardo

Apellido :

Pérez

Email :

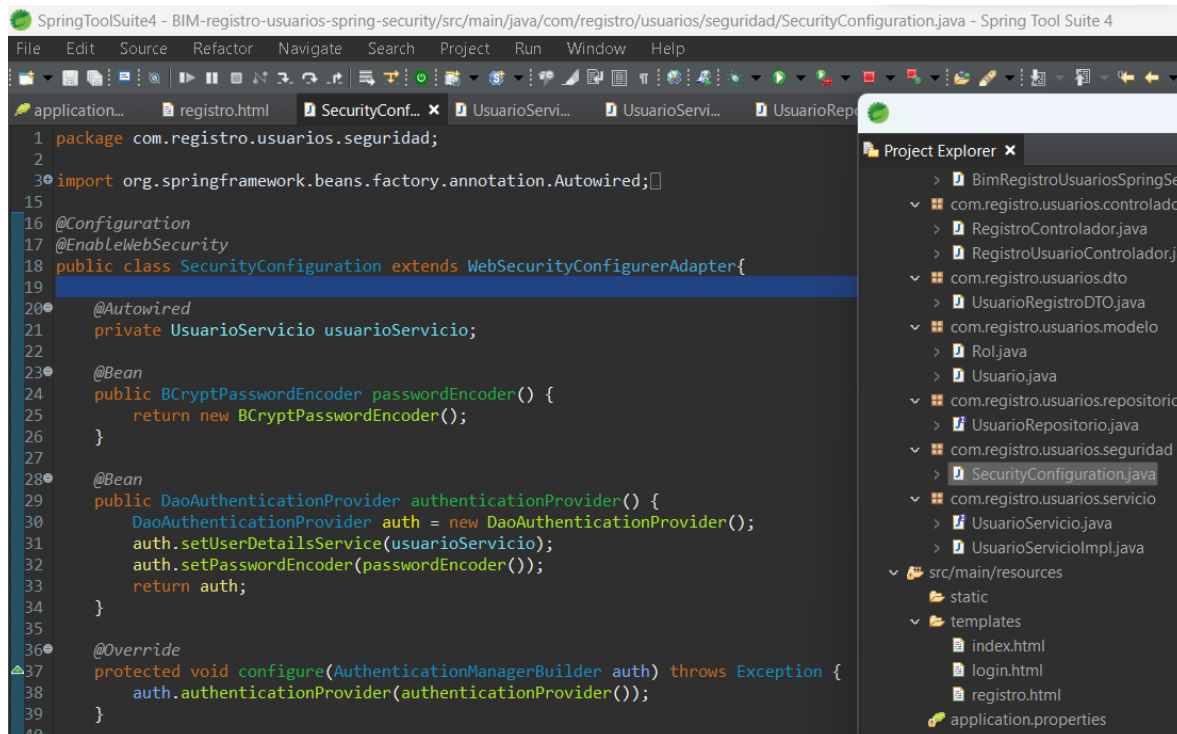
ingerar@gmail.com

Password :

....

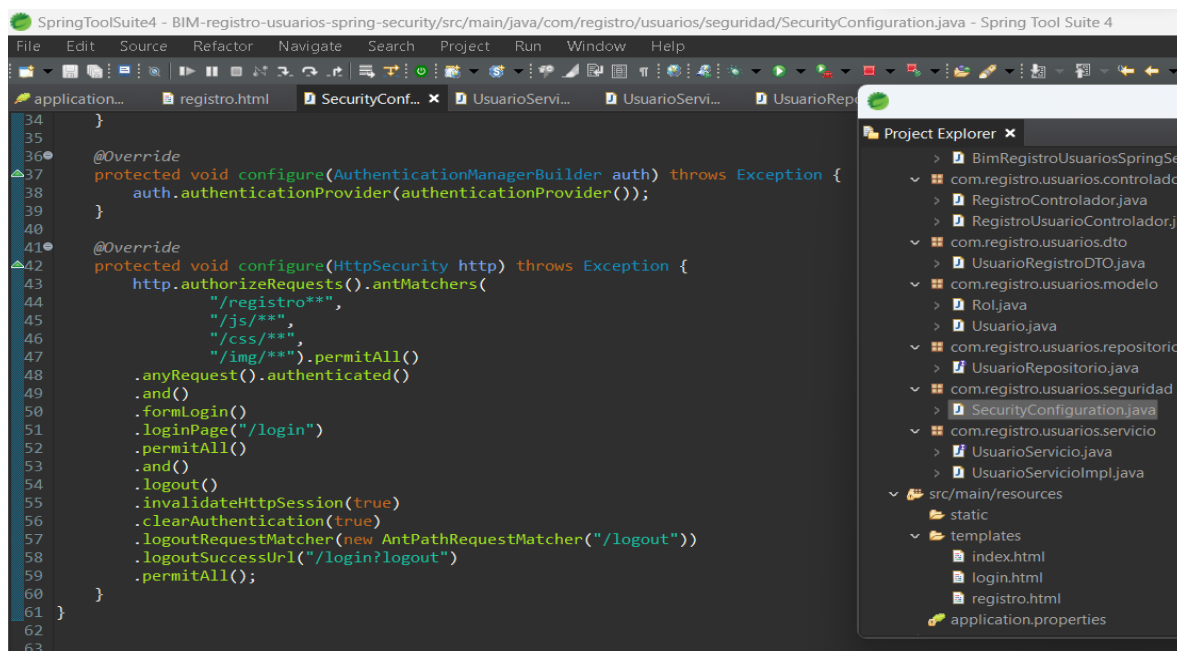
Registrar Si estas registrado inicia sesión aquí

Para iniciar ahora la parte del "login" primeramente realicé una nueva clase sobre la principal llamada "com.registro.usuarios.seguridad" y le creé una clase llamada SecurityConfiguration, la clase debe heredar WebSecurityConfigurerAdapter, pues esta clase será de configuración a la cual debe llevar sus respectivas importaciones, se debe inyectar el servicio, aquí es dónde se implementa un poco de spring security.



```
1 package com.registro.usuarios.seguridad;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5
6
7
8
9
10
11
12
13
14
15
16 @Configuration
17 @EnableWebSecurity
18 public class SecurityConfiguration extends WebSecurityConfigurerAdapter{
19
20     @Autowired
21     private UsuarioServicio usuarioServicio;
22
23     @Bean
24     public BCryptPasswordEncoder passwordEncoder() {
25         return new BCryptPasswordEncoder();
26     }
27
28     @Bean
29     public DaoAuthenticationProvider authenticationProvider() {
30         DaoAuthenticationProvider auth = new DaoAuthenticationProvider();
31         auth.setUserDetailsService(usuarioServicio);
32         auth.setPasswordEncoder(passwordEncoder());
33         return auth;
34     }
35
36     @Override
37     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
38         auth.authenticationProvider(authenticationProvider());
39     }
40 }
```

Sin olvidar que debemos permitirle las extensiones de lo que queramos agregar, debemos también crear el formulario de login y también su respectivo logout.



```
34 }
35
36 @Override
37 protected void configure(AuthenticationManagerBuilder auth) throws Exception {
38     auth.authenticationProvider(authenticationProvider());
39 }
40
41 @Override
42 protected void configure(HttpSecurity http) throws Exception {
43     http.authorizeRequests().antMatchers(
44         "/registro**",
45         "/js/**",
46         "/css/**",
47         "/img/**").permitAll()
48         .anyRequest().authenticated()
49         .and()
50         .formLogin()
51         .loginPage("/login")
52         .permitAll()
53         .and()
54         .logout()
55         .invalidateHttpSession(true)
56         .clearAuthentication(true)
57         .logoutRequestMatcher(new AntPathRequestMatcher("/logout"))
58         .logoutSuccessUrl("/login?logout")
59         .permitAll();
60 }
61 }
62
63 }
```


Creé una nueva clase dentro del paquete controlador, llamada "RegistroControlador" para agregarle los métodos correspondientes, para que cuando se ejecute la aplicación lo primero pida sea el login, colocándole sus respectivos mensajes, para que re direcciona siempre a login, sino no se avanza.

```

1 package com.registro.usuarios.controlador;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.ui.Model;
6 import org.springframework.web.bind.annotation.GetMapping;
7
8 import com.registro.usuarios.servicio.UsuarioServicio;
9
10 @Controller
11 public class RegistroControlador {
12
13     @Autowired
14     private UsuarioServicio servicio;
15
16     @GetMapping("/login")
17     public String iniciarSesion() {
18         return "login";
19     }
20
21     @GetMapping("/")
22     public String verPaginaDeInicio(Model modelo) {
23         modelo.addAttribute("usuarios", servicio.listarUsuarios());
24         return "index";
25     }
26 }

```

Ahora hice la creación del archivo login.html, para visualizar la pantalla de login correspondiente, también con las funciones de registrarse en caso de no tener una cuenta, para que el resultado final sea lo más práctico posible para el usuario final.

```

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
4 <meta charset="utf-8">
5 <title>Inicio de sesión</title>
6 <!-- Latest compiled and minified CSS -->
7 <link rel="stylesheet"
8 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
9 integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
10 crossorigin="anonymous">
11
12 <!-- Optional theme -->
13 <link rel="stylesheet"
14 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"
15 integrity="sha384-rhYoN1iRseXfVYHV4nD0JutLtnGasLCJUC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwL/Sp"
16 crossorigin="anonymous">
17 </head>
18 <body>
19
20 <nav class="navbar navbar-inverse navbar-fixed-top">
21 <div class="container">
22 <div class="navbar-header">
23 <button type="button" class="navbar-toggle collapsed"
24 data-toggle="collapse" data-target="#navbar" aria-expanded="false"
25 aria-controls="navbar">
26 <span class="sr-only">Toggle navigation</span> <span
27 class="icon-bar"></span> <span class="icon-bar"></span> <span
28 class="icon-bar"></span>
29 </div>

```

```
SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/resources/templates/login.html - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
application... registro.html SecurityConf... UsuarioServi... UsuarioServi... UsuarioRepo...
28     class="icon-bar"></span> <span class="icon-bar"></span> <span
29     class="icon-bar"></span>
30 </button>
31 <a class="navbar-brand" href="#" th:href="@{/}">Registro e
32 inicio de sesión</a>
33 </div>
34 </div>
35 </nav>
36
37 <br>
38 <br>
39
40 <div class="container">
41 <div class="row">
42 <div class="col-md-6 col-md-offset-3">
43 <h1>Inicio de sesión</h1>
44 <form th:action="@{/login}" method="post">
45
46 <div th:if="${param.error}>
47 <div class="alert alert-danger">Usuario o contraseña
48 inválidos</div>
49 </div>
50
51 <div th:if="${param.logout}>
52 <div class="alert alert-info">Ha cerrado sesión exitosamente
53 </div>
54 </div>
55
56
57 <div class="form-group">
```

```
SpringToolSuite4 - BIM-registro-usuarios-spring-security/src/main/resources/templates/login.html - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
application... registro.html SecurityConf... UsuarioServi... UsuarioServi... UsuarioRepo... RegistroCon... login.html x UsuarioRegi...
57 <div class="form-group">
58 <label for="username">Nombre de
59 usuario : </label> <input id="username" name="username" type="text" class="form-control"
60 required autofocus="autofocus" placeholder="Digite su email ID">
61 </div>
62
63 <div class="form-group">
64 <label for="password">Password : </label> <input
65 id="password" type="password" name="password" class="form-control" required
66 autofocus="autofocus" placeholder="Digite su password">
67 </div>
68
69 <div class="form-group">
70 <div class="row">
71 <div class="col-md-6 col-md-offset-3">
72 <input type="submit" class="form-control btn btn-primary" name="Login-submit" id="Login-submit"
73 value="Iniciar sesión" />
74 </div>
75 </div>
76 </div>
77 </form>
78 <div class="form-group">
79 <span>Si eres nuevo usuario <a th:href="@{/registro}">regístrate
80 aqui</a></span>
81 </div>
82
83 </div>
84 </div>
85 </div>
```



```

85     </div>
86
87
88     <!-- Latest compiled and minified JavaScript -->
89     <script
90         src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
91         integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7L2mCWNIpG9mGCD8wGNIcPD7Txa"
92         crossorigin="anonymous"></script>
93 </body>
94 </html>

```

VISUALIZACIÓN DEL SEGUNDO HTML (LOGIN)

La pantalla de login debe contener el mensaje de alerta cuando el login es incorrecto o bien no existe en la base de datos (no se ha registrado) además de colocar text fields en los campos para mayor comodidad de qué datos se deben agregar y dónde por mano del usuario final, también con la opción configurada de crear cuenta nueva en caso de no tener alguna hecha.

Registro e inicio de sesión

Inicio de sesión

Usuario o contraseña inválidos

Nombre de usuario :

Digite su email ID

Password :

Digite su password

Iniciar sesión

Si eres nuevo usuario [regístrate aquí](#)

VISUALIZACIÓN DEL TERCER HTML PARA EL FRONT COMPLETO (INDEX)

Para finalizar creé el index, el cual mostrará los datos almacenados siempre y cuando hayan podido hacer login, mostrando los datos de ID, nombre, apellido y el email, pues la contraseña no debe mostrarse debido a que pueden obtenerla los demás usuarios, además con el método de password encode, hace que sea imposible visualizarla aún para el administrador de la base de datos, sin embargo está la opción de eliminar dicha cuenta de la base de datos para volverla a crear en el caso la contraseña sea olvidada por el usuario, codificándola de ese modo.

```

1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org"
3     xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
4 <head>
5 <meta charset="utf-8">
6 <title>Inicio</title>
7 <!-- Latest compiled and minified CSS -->
8 <link rel="stylesheet"
9     href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
10     integrity="sha384-BVYiiSIFeK1dGmJRAKycuHAHRg320mUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
11     crossorigin="anonymous">
12
13 <!-- Optional theme -->
14 <link rel="stylesheet"
15     href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"
16     integrity="sha384-rHyoN1iRsVXV4nD0JutLNgasLCJuC7uwjduW9SVrLvRYooPp2bWYmgJQIXwL/Sp"
17     crossorigin="anonymous">
18 </head>
19 <body>
20
21
22 <nav class="navbar navbar-inverse navbar-fixed-top">
23 <div class="container">
24 <div class="navbar-header">
25 <button type="button" class="navbar-toggle collapsed"
26     data-toggle="collapse" data-target="#navbar" aria-expanded="false"
27     aria-controls="navbar">
28 <span class="sr-only">Toggle navigation</span> <span
29     class="icon-bar"></span> <span class="icon-bar"></span> <span
30     class="icon-bar"></span>

```

```

29 <span class="icon-bar"></span> <span class="icon-bar"></span> <span
30 <span class="icon-bar"></span>
31 </button>
32 <a class="navbar-brand" href="#" th:href="@{/}">Registro e
33     inicio de sesión</a>
34 </div>
35 <div id="navbar" class="collapse navbar-collapse">
36 <ul class="nav navbar-nav">
37 <li sec:authorize="isAuthenticated()"><a th:href="@{/logout}">Cerrar
38     sesión</a></li>
39 </ul>
40 </div>
41 </div>
42 </nav>
43
44 <br>
45 <br>
46
47 <div class="container">
48 <h1>Registro e inicio de sesión con Spring Boot + Spring Security
49     + Thymeleaf + Hibernate + MySQL y Bootstrap</h1>
50 <br>
51 <h4>
52     Bienvenido <span sec:authentication="principal.username"></span>
53 </h4>
54
55 <br> <br>
56
57 <table class="table table-striped table-bordered">
58 <thead class="table-dark">

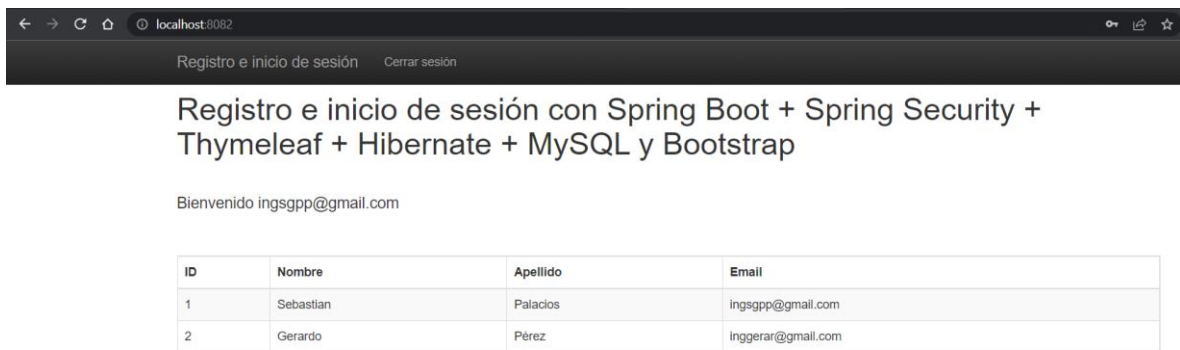
```

```

50
57●   <table class="table table-striped table-bordered">
58●       <thead class="table-dark">
59●           <tr>
60               <th>ID</th>
61               <th>Nombre</th>
62               <th>Apellido</th>
63               <th>Email</th>
64           </tr>
65       </thead>
66●       <tbody>
67●           <tr th:each="usuario : ${usuarios}">
68               <td th:text="${usuario.id}">ID</td>
69               <td th:text="${usuario.nombre}">ID</td>
70               <td th:text="${usuario.apellido}">ID</td>
71               <td th:text="${usuario.email}">ID</td>
72           </tr>
73       </tbody>
74   </table>
75 </div>
76
77
78
79   <!-- Latest compiled and minified JavaScript -->
80●   <script
81       src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
82       integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7L2mCWNIpG9mGCD8wGNiCPD7Txa"
83       crossorigin="anonymous"></script>
84 </body>
85 </html>

```

Así es como culminé el proyecto, la pantalla final mostrará los resultados sugeridos de 4 campos ocultando la contraseña, la interfaz incluye también la opción de cerrar sesión.



Registro e inicio de sesión con Spring Boot + Spring Security + Thymeleaf + Hibernate + MySQL y Bootstrap

Bienvenido ingsgpp@gmail.com

ID	Nombre	Apellido	Email
1	Sebastian	Palacios	ingsgpp@gmail.com
2	Gerardo	Pérez	inggerar@gmail.com