In a previous post we had implemented Spring Boot Security – Database authentication using JDBC.

We authenticated users and roles against database tables. But the entries for users and the roles in the table were added using script file. In this Use case we will see how this can be done programmatically.

1. Continue with ***“boot-security-loginwith-users”***

add the model class for user registration.

```
package com.demo.model;

public class UserRegistration {

	private String username;
	private String password;

	public String getUsername() {
		return username;
	}

	public void setUsername(String username) {
		this.username = username;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

}
```

2.

Next add the controller methods for the user registration.
Using the GET method we return the registration page.

Using the Spring form tag with the modelAttribute we specify the backing bean for this form.

Each input field we make use of the form:input tag, this automatically binds the field value to the corresponding value of the backing bean object.

When the user clicks submit the POST method call is made to the controller and the form is automatically bound to the user argument we passed.
Using the bounded user object we then create the database entries using ***JdbcUserDetailsManager***.

JdbcUserDetailsManager provides CRUD operations for both users and authorities granted.

EmployeeController.java

```
@RequestMapping(value = "/register", method = RequestMethod.GET)
	public ModelAndView register() {
		return new ModelAndView("registration", "user", new
UserRegistration());
	}

	@RequestMapping(value = "/register", method =
RequestMethod.POST)
	public ModelAndView processRegister(@ModelAttribute("user")
UserRegistration userRegistrationObject) {

		// authorities to be granted
		List<GrantedAuthority> authorities = new
ArrayList<GrantedAuthority>();
		authorities.add(new SimpleGrantedAuthority("ROLE_ADMIN"));

		User user = new User(userRegistrationObject.getUsername(),
userRegistrationObject.getPassword(), authorities);
		jdbcUserDetailsManager.createUser(user);
		return new ModelAndView("redirect:/welcome");
	}
```

3.

Next we modify the security configuration to-

- Create a bean of type JdbcUserDetailsManager
- allow /register page without any security

3.1 EmployeeSecurityConfiguration.java

```
@Bean
	public JdbcUserDetailsManager jdbcUserDetailsManager()
throws Exception {
		JdbcUserDetailsManager jdbcUserDetailsManager = new
JdbcUserDetailsManager();
		jdbcUserDetailsManager.setDataSource(dataSource);
		return jdbcUserDetailsManager;
	}
```

3.2. This is the only change but rewrite this method if you want to

```
http.authorizeRequests().antMatchers("/register").permitAll()
```

```
@Override
     protected void configure(HttpSecurity http) throws Exception {
          http.authorizeRequests().antMatchers("/
register").permitAll().antMatchers("/welcome").hasAnyRole("USER",
"ADMIN")
                    .antMatchers("/getEmployees").hasAnyRole("USER",
"ADMIN").antMatchers("/addNewEmployee")
                    .hasAnyRole("ADMIN").anyRequest().authenticated(
).and().formLogin().loginPage("/login").permitAll()
                    .and().logout().permitAll();

          http.csrf().disable();
     }
```

4. Next we will create a new registration page with modelAttribute tag.

registration.jsp

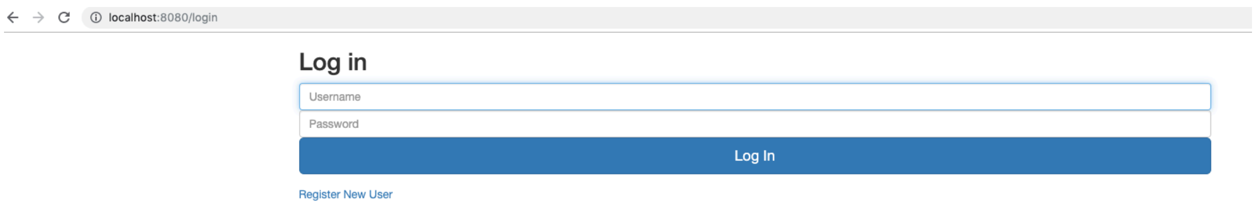
```
<%@taglib uri="http://www.springframework.org/tags/form"
prefix="form"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
<title>Register User</title>
</head>
<body>
     <h3 style="color: red;">Register New User</h3>

     <div id="registerEmployee">
          <form:form action="/register" method="post"
modelAttribute="user">
               <p>
                    <label>Enter username</label>
                    <form:input path="username" />
               </p>
               <p>
                    <label>Enter password</label>
                    <form:input path="password" />
               </p>
               <input type="SUBMIT" value="Submit" />
          </form:form>
     </div>
</body>
</html>
```

5. Add the Register link in the login.jsp page

after <form> tag.

```
<a href="/register">Register New
               User</a>
```

6. OUTPUT



7. Click on “Register New User” and add new ADMIN ROLE User.

8. Login as the new user and check