Date : 12-Jun-21
Spring Boot 9AM
Mr. RAGHU

Spring Security using ORM

1. User Register Process [done]
2. User Login and PasswordEncoder [done]
3. Custom Login Page [done]
4. Session Management View
5. CSRF View

Stage # 3. Custom Login Page

=> Spring Security has provided one pre-defined Login Page.
=> If we enter /login with GET type, you can see that login page
=> on click submit , request sent with /login + POST type.

--sample HTML Code for Login Form looks like---

```
<form  method="post" action="/login">
 <input type="text" id="username" name="username"/>
 <input type="password" id="password" name="password"/>
 <button type="submit">Sign in</button>
</form>
```

On Login Failed: (error parameter exist)
http://localhost:8081/login?error (default)

http://localhost:8081/user/login?error (new value)

On Logout Success: (logout parameter exist)
http://localhost:8081/login?logout (default)

http://localhost:8081/user/login?logout (new value)

*) Note:
a. We can define method to show login page. (/login, GET)
b. We can not define method to vailidate data, it is already exist
     [Validate : UsernamePasswordAuthenticationFilter,
        to create/read session: SessionFixationProtectionStrategy]
        fixed: /login, POST

-----coding steps-----

UserLogin.html

```
<form  method="post" action="/login">
  <input type="text" id="username" name="username"/>
  <input type="password" id="password" name="password"/>
  <button type="submit">Sign in</button>
</form>
```

UserLogin.html(with UI Design)

```
<html xmlns:th="https://www.thymeleaf.org/">
<head>
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.
```

```
css"/>
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" >
</script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.
min.js" ></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js
"></script>
</head>
<body>
        <div class="container">
                <div class="card">
                        <div class="card-header bg-primary text-white
text-center">
                                <h3>USER LOGIN PAGE!</h3>
                        </div>
                        <div class="card-body">
                                <form th:action="@{/login}"
method="POST">

                                        <div class="row">
                                                <div class="col-3">

<label>EMAIL</label>
                                                </div>
                                                <div class="col-6">
                                                        <input
type="text" name="username" id="username" class="form-control"
required/>
                                                </div>
                                        </div>

                                        <div class="row">
                                                <div class="col-3">

<label>PASSWORD</label>
                                                </div>
                                                <div class="col-6">
                                                        <input
type="password" id="password" name="password" class="form-control"
required/>
                                                </div>
                                        </div>

                                         <button type="submit"
class="btn btn-success">Sign in</button>
                                </form>
                        </div>
                        <div th:if="${param.error}" class="card-footer
bg-danger text-white">
                                Username/Password Invalid.
                        </div>
                        <div th:if="${param.logout}" class="card-footer
bg-info text-white">
                                User logged-out Succesfully!
                        </div>
```

```
                </div>
        </div>

        </body>
</html>
```

2. Specify method in controller such that it will show login page

```
@Controller
@RequestMapping("/user")
public class UserController {
         //3. show login page
        @GetMapping("/login")
        public String showLogin() {
                return "UserLogin";
        }
}
```

4. This URL can be accessed by everyone

```
.antMatchers("/user/login").permitAll()
```

5. Modify Default Login , default values

	Default Values	NewValues
ShowLogin Page	/login GET	/user/login GET
Login failed	/login?error	/user/login?error
LogoutSuccess	/login?logout	/user/login?logout
LoginProcessing	/login POST	/login , POST

```
                                        <form  th:action="@{/login}"
method="POST"> </form>
```

```
****SecurityConfig************************
package in.nareshit.raghu.config;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.builders.A
uthenticationManagerBuilder;
import
org.springframework.security.config.annotation.web.builders.HttpSecurity
;
import
org.springframework.security.config.annotation.web.configuration.EnableW
ebSecurity;
import
org.springframework.security.config.annotation.web.configuration.WebSecu
rityConfigurerAdapter;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import
org.springframework.security.web.util.matcher.AntPathRequestMatcher;
```

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

	@Autowired
	private UserDetailsService userDetailsService;

	@Autowired
	private BCryptPasswordEncoder passwordEncoder;


	protected void configure(AuthenticationManagerBuilder auth)
throws Exception {
		auth
		.userDetailsService(userDetailsService)
		.passwordEncoder(passwordEncoder);
	}

	protected void configure(HttpSecurity http) throws Exception {
		http.authorizeRequests()

.antMatchers("/user/register","/user/save","/user/login").permitAll()
		.antMatchers("/home").permitAll()
		.antMatchers("/admin").hasAuthority("ADMIN")
		.antMatchers("/emp").hasAnyAuthority("ADMIN","EMPLOYEE")
		.anyRequest().authenticated()

		.and()
		.formLogin()
		.loginPage("/user/login")  //def: /login, GET
		.loginProcessingUrl("/login")
		.defaultSuccessUrl("/profile",true)
		.failureUrl("/user/login?error")//def: /login?error

		.and()
		.logout()
		.logoutRequestMatcher(new
AntPathRequestMatcher("/signout"))
		.logoutSuccessUrl("/user/login?logout") //def: /login?
logout

		.and()
		.exceptionHandling()
		.accessDeniedPage("/denied")
		;
	}
}
```

Cloud Deploy:

a. Register : https://signup.heroku.com/login

b. Validate Email (Goto Mail inbox)

c. Login
 https://id.heroku.com/login

d. Enter un/pwd > click on login

----(code modifications)----------

Link with Postgress

--application.properties--
server.port=9900

spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://localhost:5432/postgres
spring.datasource.username=postgres
spring.datasource.password=root

spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=create
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL10Dialect
spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true
---

pom.xml
a. comment MySQL Dependency
b. add postgress dependency
> Right click on project > Spring > Add Starters> Choose Postgress

```
<dependency>
        <groupId>org.postgresql</groupId>
        <artifactId>postgresql</artifactId>
        <scope>runtime</scope>
</dependency>
```

----(Github)----
a. Register, Email validation, login
b. create one repository and push local code to gihub
https://www.youtube.com/watch?v=T2UHpsxJ-2o

-----(heroku)----
a. login
b. enter URL: https://dashboard.heroku.com/apps
c. click on create new app
 Enter name
 ex: spring-boot-security-sample
 > create App

d. Link with github
> Goto Deploy
> Goto Deployment Methods
> Click on git hub
> First time : username/password
> Enter repository name
> click on connect
> Deploy branch

https://github.com/javabyraghu/SpringBoot2SecurityOrmEx.git