LAB- Spring Security Registration , Email Verification and Remember me

Step 1 – User Registration

In this step you will be working on **02-security-registration-start** project.

We want a registration page for registering application users.

Open **register.html** under **src/main/resources/templates** folder and observe the fields in it.

We need a DTO corresponding to the fields in html.

Open Schema.sql and observe the ddl for users and authorities table.

We need Entities to represent these tables.

In micro-lab-docs folder, I have given a folder with name **security-registration-code.** In this folder there are some class files. When ever I ask you to copy files, u should copy from this folder.

Copy MyAppUser.java from com/way2learnonline/model folder into com.way2learnonline.model package in **02-security-registration-start** project and observe the jpa mappings

Copy UserRepository.java from com/way2learnonline/repository folder into com.way2learnonline.repository package and observe the code.

Copy UserRegistrationService.java and UserRegistrationServiceImpl.java from com/way2learnonline/service folder into com.way2learnonline.service package and observe the code

We need a DTO to capture the data coming from register.html.

Copy UserDto.java from com/way2learnonline/model folder into com.way2learnonline.model package in **02-security-registration-start** project and observe the annotations to validate. You will see some error in this UserDto.java. Don't worry as of now. Proceed to next step.

Some custom annotations are used for validating the data.

All the validators are kept in com/way2learnonline/validation folder. Copy them into com.way2learnonline.validation package. I have already explained you how these validators work.

Now, errors in UserDto.java should disappear

Copy RegistrationController.java from com/way2learnonline/controllers folder into com.way2learnonline.controllers package and observe the code. I have explained you already how validations are working because of @Valid annotation

Open mylogin.html and observe that there is a link to register.html

Now start this on **02-security-registration-start** application from boot dash board.

Go to http://localhost:8080/h2-console and observe that Users and authorities tables are populated with data from data.sql and there are 3 users.

Now give a request to http://localhost:8080/hello?name=Siva
You will be redirected to login page. Click on Register Link in this page to see the registration page.

Give details and register a user.
After registration is successful, observe the users and authorities table and make sure that user registration details are saved in database.

Try to login with new user. Your login should be successful.

Step 2 – Email Verification

When a new User is registered, we want to disable that user until email verification is done.

Open RegistrationController.java and modify the MyAppUser creation as shown below to disable the user.

```
MyAppUser myuser = new MyAppUser(user.getUsername(),
                        user.getFirstname(),
                        user.getLastname(),
                        user.getEmail(),
                        encoder.encode(user.getPassword()),false);
```

When ever a user is registered we want to raise our own custom event.

Copy UserRegistrationEvent.java from com/way2learnonline/events folder into com.way2learnonline.events package and observe the code

In Registration Controller, complete TODO-15 , TODO-16 and TODO-17

We want to write a Event Listener which creates a unique verification id mapped to username ans store it in db.

When Verification request comes, verificationid will be sent. We have to get the mapped username and enable that user.

So, we need a table to store the verification id and username mapping.

Copy Verification.java into com.way2learnonline.model package and observe that it is mapped to an Entity

Copy VerificationCodeRepository.java into com.way2learnonline.repository package and observe the code in it.

Copy VerificationService.java into com.way2learnonline,service package and observe the code in it.

Now copy EmailVerificationListener.java into com.way2learnonline,service package and observe the code in it.

Now Copy VerificationController.java into com.way2learnonline.controller package and observe the code in it.

To send a mail using gmail we need app password. Go to the below link to generate app password for your gmail id.

https://support.google.com/accounts/answer/185833?p=InvalidSecondFactor

Open application.yml and configure following:

```
spring:
  mail:
    host: smtp.gmail.com
    port: 587
    username: sivaprasad.valluru@gmail.com
    password: yourapppassword
    protocol: smtp
    test-connection: false
    properties:
      mail:
        smtp:
```

```
    auth: true
    starttls:
      enable: true
```

Now restart your application and register a new user.
Try logging in using this username/password. Your login should fail as user is not enabled.

Check your mail and activate the user.

Now you should be able to login

Step 3 – Remember me

We want to enable remember me functionality for my app.
Open login.html and uncomment the check box code for remember me check box

Open WebSecurityConfiguration.java and enable rememberme filter as shown below:

```
http
.authorizeRequests()
.antMatchers("/register","/login","/h2-console/**","/mylogin","/verify/**").permitAll()
    .anyRequest().authenticated()
    .and()
    .formLogin().loginPage("/login").defaultSuccessUrl("/hello", true)
    .and().csrf().disable().rememberMe().key("myremembermekey")
    ;
```

Now, give a request to http://localhost:8080/hello?name=Siva

You should be redirected to login page and now you should see remember me check box.

Press F12 on your firefox browser and click on storage tab to see cookies. You should see only jsessionid cookie.

Now submit your credentials and login.
After logging in, you should see remember-me cookie also.

Now just delete the jsessionid cookie and refresh the browser. You will not be prompted for login again. You will be automatically authenticated using remember me cookie

I have explained you how

In WebSecurityConfiguration.java, configure logout such that remember-me cookie is deleted once logout is successful as shown below:

```
http
.authorizeRequests()
.antMatchers("/register","/login","/h2-console/**","/mylogin","/verify/**").permitAll()
    .anyRequest().authenticated()
    .and()
    .formLogin().loginPage("/login").defaultSuccessUrl("/hello", true)
    .and().csrf().disable().rememberMe().key("myremembermekey")

    .and().logout().logoutUrl("/logout").logoutSuccessUrl("/login").deleteCookies("remember-me");
```

If you click on logout link, observe that remember-me cookie is also deleted

CONGRATULATIONS YOU KNOW HOW TO DO USER REGISTRATION , add email verification and add remember me functionality