=>using InMemory DB (RAM Level DB) as authentication pprovider is good only in Dev,Test env../profile of the Project.. In Uat, Prod env../profile prefer using DB s/w or LDAP Server as the authentication provider using

Procedure to develope spring boot security App that is InMemory DB (RAM Level) as the authetication provider

url

jsp page |---> home.jsp

security enabled

permitAll()

DB

(DB be will be created on the

startup of application and DB

will be vanished on shutdown of App)

/offers offers.jsp

show

/balance balance.jsp

/loanApprove loan.jsp;

authenticated()

authenticated() + Authorization() :: hasAnyRole("CUSTOMER","MANAGER") authenticated + authrozation :: haŝRole("MANAGER")

boot

=>once we add spring security starter to spring web mvc/sping rest/ spring Ms project then it automatically takes care of configuring DelagatingFilterProxy Servlet Filter with "/" url pattern

step1) create spring stater project adding the following starters

(The changes will be updated automatically)

a) spring security b) spring web c) spring devtools

step2) Develop the regular controller class having different handler methods with different request paths

controller class

==========

```
package com.nt.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
@Controller
public class BankOperationsController {
@GetMapping("/")
public String showHome() {
return "home";
}
```

jsp pages

======

home.jsp

(choose the version 2.7.6 in pom.xml)

[Configure and run the application in Tomcat 9.x]

Spring boot 2.x is compitable with tomcat 9.x (javax.servlet pgks) Spring boot 3.x is compitable with tomcat 10.x (Jakarta.servlet pkgs)

```
<%@page isELIgnored="false"%>
@GetMapping("/offers")
public String showOffers() { return "offers";
}
@GetMapping("/balance")
public String checkBalnace() { return "show_balance";
}
@GetMapping("/loanApprove")
public String approveLoan() { return "loan";
}
@GetMapping("/denied")
public String accessDenied() { return "access denied";
}
<h1 style="color:red;text-align:center"> Welcome Xyz Bank --Home page </h1>
<a href="balance"> Check Balance</a> <br> <br>
<a href="loanApprove"> Approve Loan</a> <br> <br>
<a href="offers"> Show offers</a> <br>
```

//loan.jsp

```
<%@ page isELIgnored="false" import="java.util.*"%> <!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1 style="color:blue;text-align:center"> Loan Approval Page </h1>
<b>u rapproved for loan amount :: <%=new Random().nextInt(1000000) %> <a href="./">Home</a>
Handler method to show authorization failure page
</body>
</html>
```

show_balance.jsp

=====

```
<%@ page isELIgnored="false" import="java.util.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1 style="color:blue;text-align:center"> ShowBalance Page </h1> <b> balance :: <%= new Random().nextInt(100000) %></b>
<a href="./">Home</a>
```

offers.jsp

=======

```
<%@ page isELIgnored="false"%> <!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1 style="color:blue;text-align:center"> Show Offers Page </h1>
Home Loan ROI :: 7% <br>
FourWheeler Loan ROI :: 8% <br>
Personal Loan ROI: 12% <br>
<a href="./">Home</a>
</body> </html>
```

step3)

/ ---> permitAll()

```
</body> </html>
```

Decide authentication and authorization level for different requeest urls

/offers --> authenticated()

/balance ---> authenticated() + Authorization() :: hasAnyRole("CUSTOMER", "MANAGER") /loanApprove ---> authenticated + authrozation :: hasRole("MANAGER")

step4) Develope SecurityConfig class extending WebSecurityConfigurerAdapter and having annotations @Configuration + @EnableWebSecurity and also overriding two configure(-) methods

To cfg

SEcurityConfig.java

(This Configuration class will be linked with main configuration class(@SpringBootApplication) internally)

```
package com.nt.config;

import org.springframework.context.annotation.Configuration;

import
org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

@Configuration

@EnableWebSecurity // makes the Normal @Configuration class Spring Security Configuration class public
class SecurityConfig extends WebSecurityConfigurerAdapter {

Authentication Provider

@Override

deprecated from spring security 5.4

public void configure (AuthenticationManagerBuilder auth) throws Exception {

// Build Authentication Manager by taking given Authentication Info Provider (InMemoryDB) /*
auth.inMemoryAuthentication().withUser("raja").password("{noop}rani").authorities("CUSTOMER");

auth.inMemoryAuthentication().withUser("ramesh").password("{noop}hyd").authorities("MANAGER");*/

auth.inMemoryAuthentication().withUser("raja").password("{noop}rani").roles("CUSTOMER");
auth.inMemoryAuthentication().withUser("ramesh").password("{noop}hyd").roles("MANAGER");

Creates InMemory DB (RAM level DB)

and uses it a Authentication Provider

@Override

Actually we need to pass password as the encrypted password.. if do not want pass, pass it as {noop} <pwd>
which indicates NoOperationEncoder is used for Encryption

public void configure(HttpSecurity http) throws Exception { //authorize requests

http.authorizeRequests().antMatchers("/").permitAll() //Not authetication an no authorization
.antMatchers("/offers").authenticated() //only authentication

.antMatchers("/balance").hasAnyRole("CUSTOMER","MANAGER") // authentication + authorization for
"CUSTOMER","MANAGER" role uses .antMatchers("/loan Approve").hasRole("MANAGER") //// authentication
+ authorization for .anyRequest().authenticated() //remaing all requests url mus be authtenticated

Authentication +

Authorizat on cfgs

//specify authentication mode (Uses the browser managed dialog for collecting username, passsword from
enduser) .and().httpBasic()

}

}

//exception/error handling (for 403 error) .and().exceptionHandling().accessDeniedPage("/denied");

This request path based handler method
```

of Controller class executes to display the

error page for authrization failure.

Do following operation two operations to cfg

custom error page for 403 error

1) add handler method in the controller class with

request path "/denied"

```
@GetMapping("/denied")
public String showAccessDienedPage() {
}
return "authorization_failed";
```

2) palce authorization_failed.jsp page in src/main/webapp/pages folder

step5) develop the application.properties having required entries

application.properties

```
# Embedded Tomcat server port number
server.port=4041
#View Resolver cfg
spring.mvc.view.prefix=/WEB-INF/pages/
spring.mvc.view.suffix=.jsp
```

step5) Run the Application (we can use embeded Tomcat or extenal Tomcat)

note:: while runnung the above web application as spring boot app using embedeed Tomcat server, we need to add tomcat embedded jasper dependency

```
<!-- https://mvnrepository.com/artifact/org.apache.tomcat.embed/tomcat-embed-jasper --> <dependency>
<groupId>org.apache.tomcat.embed</groupId>
<artifactId>tomcat-embed-jasper</artifactId>
</dependency>
```

BootSecurityProj01-SecurityApp1 [boot] [devtools]

>Spring Elements

#src/main/java

✓com.nt

> BootSecurity Proj01SecurityApp1Application.java

com.nt.config

>WebSecurityConfig.java

com.nt.controller

> BankOperationsController.java

src/main/resources

`static`

templates

application.properties

> #src/test/java

> JRE System Library [JavaSE-11]

> Maven Dependencies

✓ src

✓ main

authorization_failed.jsp

>Authorization failed</h1>

> java

› resources

✓ webapp

WEB-INF

✓ pages

authorization_failed.jsp

home.jsp

loan_approve.jsp show_balance.jsp

show_offers.jsp

>test

> target

WHELP.md mvnw

mvnw.cmd

Limitations of BASIC mode authentication

box

a) The dialog box asking username, password is browser specific dialog and it can not be customized

b) Does not allow to add the following features

a) Logout b) rembermber Me c) SessionMaxActiveCount Limit and etc...

=>To overcome the above problems take the support of form login

=>

.and().httpBasic() ---> gives BASIC mode of authetication

.and().formLogin() --> gives FORM mode of mode of Authetication. (we can use readymade forms or custom forms)

.and().rememberMe()

=>Adds another Filter supporting Remember me Authentication.. Internally uses Persistent cookies to remember given username, password having 48 hour expiry time.. In this 48 hours

after successful singin .. if u close the browser by taking the URL from browser address bar .. then

we can use the same url to get back to the page with out any signin activitty.

77 add SessionMaxConcurrency count

praticals :: launch app --> open home page --> (select remember me) signin using any correct username,

password ---> close browser wndow with out signout by copying the URL --> relauch the same browser by pasting the URL supprisingly we wil not ask username, password again

_.and().sessionManagement().maximumSessions(2).maxSessions Prevents Login(true); (place this line as the

=>Adds another Filter to controller max sessions for each user to operate the application.

//add Logout Filter

.and().logout()

code in UI page

logout

last line in the method chaining)

==>adds another Filter providing signout activity having the url or request path "/logout".. by default This can be

changeddtional code.

can be with

.and().logout().logoutRequestMatcher(new AntPathRequestMatcher("/signout"))

Code in UI page

logout

Internal of Session Management

if Login is successful it creates new Session::

HttpSession ses=req.getSession(); or HttpSession ses=req.getSession(true);

To access the existing session

HttpSession ses=req.getSession(false);

To stop/invalidate the Session

ses.invalidate();

To specify max inactive interval period for a Session

30

ses.setMaxInactiveInterval(20); //20 mins //default is mins

How to configure custom Login form page for Spring Boot Security app

==========

==========

=>For this we need call more methods on the top formLogin() method chain
.and().formLogin().loginPage("showLogin")

It is the handler method request path using which

we want to launch the form page

↓

In the user-defined form page, we need to take the following

fixed names

a) request mode :: POST

b) action url:: login

c) text box name :: username

d) password box name :: password

BootSec Workspace nMemoryDB [boot] [devtools]

> u. Deployment Descriptor: BootSecurity01-Basic-InMemoryDB Spring Elements

›

>JAX-WS Web Services

#src/main/java

#com.nt

› BootSecurity01 BasicInMemoryDbApplication.java

> ServletInitializer.java

✓com.nt.config

>SecurityConfig.java

com.nt.controller

> BankOperationsController.java

> src/main/resources

>src/test/java

> JRE System Library [JavaSE-11]

Maven Dependencies

> Deployed Resources

src

ਆ ਕਿ main

✓ java

com

> resources

webapp WEB-INF ✓ pages

access_denied.jsp home.jsp loan.jsp offers.jsp

show_balance.jsp

> test

> target

WHELP.md

mvnw

mvnw.cmd Mpom.xml