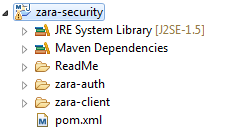
OAUTH2 with 3 Legged Authentication – InMemory Based

# **Project Structure**



In this project , you can find two war file, one for authorization server and another is client. The authorization is also the api server, it can provide oauth2 token and also provide the response by making rest call. The client application is simply used to receive the token and enable you to get the response from the server.

# **Maven build(parent pom.xml)**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>zara-security</groupId>

<artifactId>zara-security</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>pom</packaging>

<name>zara-security</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<spring.version>3.2.1.RELEASE</spring.version>

<spring.security.version>3.1.3.RELEASE</spring.security.version>

<spring.security.oauth2.version>1.0.2.RELEASE</spring.security.oauth2.version>

</properties>

<modules>

<module>zara-auth</module>

<module>zara-client</module>

</modules>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-web</artifactId>

<version>${spring.security.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-core</artifactId>

<version>${spring.security.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-config</artifactId>

<version>${spring.security.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-taglibs</artifactId>

<version>${spring.security.version}</version>

</dependency>

<dependency>

<groupId>org.springframework.security.oauth</groupId>

<artifactId>spring-security-oauth2</artifactId>

<version>${spring.security.oauth2.version}</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jcl-over-slf4j</artifactId>

<version>1.6.6</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.6.6</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.0.9</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.21</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.3</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

</configuration>

</plugin>

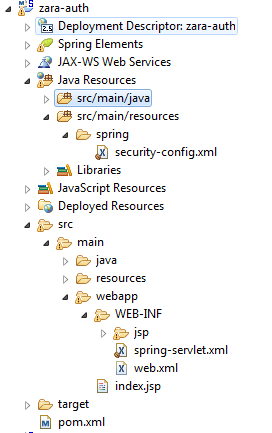
</plugins>

</build>

</project>

Let us move to the first authorization server details.

# **Auth server Project Structure**



# **Maven Configuration**

<?xml version=*"1.0"*?>

<project xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"* xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>zara-security</groupId>

<artifactId>zara-security</artifactId>

<version>0.0.1-SNAPSHOT</version>

</parent>

<groupId>zara-auth</groupId>

<artifactId>zara-auth</artifactId>

<packaging>war</packaging>

<name>zara-auth Maven Webapp</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<finalName>zara-auth</finalName>

</build>

</project>

# **Spring Security Configuration(resources/spring/security-config.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns:aop=*"http://www.springframework.org/schema/aop"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"*

xmlns:oauth2=*"http://www.springframework.org/schema/security/oauth2"*

xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:security=*"http://www.springframework.org/schema/security"*

xmlns:tx=*"http://www.springframework.org/schema/tx"*

xmlns:mvc=*"http://www.springframework.org/schema/mvc"*

xsi:schemaLocation=*"http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-3.1.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.1.xsd*

*http://www.springframework.org/schema/security/oauth2 http://www.springframework.org/schema/security/spring-security-oauth2.xsd*

*http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd*

*http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security-3.1.xsd*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.1.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.1.xsd"*>

<context:component-scan base-package=*"com.ddlab.rnd.security.\*"*/>

<mvc:default-servlet-handler />

<security:http pattern=*"/test/\*\*"* create-session=*"never"*

entry-point-ref=*"oauthAuthenticationEntryPoint"*

access-decision-manager-ref=*"accessDecisionManager"*>

<security:anonymous enabled=*"false"* />

<security:intercept-url pattern=*"/test/test/\*\*"*

access=*"ROLE\_USER"* />

<security:custom-filter ref=*"resourceServerFilter"*

before=*"PRE\_AUTH\_FILTER"* />

<security:access-denied-handler ref=*"oauthAccessDeniedHandler"* />

</security:http>

<oauth2:resource-server id=*"resourceServerFilter"*

token-services-ref=*"tokenServices"* />

<security:http pattern=*"/oauth/token"* create-session=*"stateless"*

authentication-manager-ref=*"clientAuthenticationManager"*

entry-point-ref=*"oauthAuthenticationEntryPoint"*>

<security:intercept-url pattern=*"/oauth/token"*

access=*"IS\_AUTHENTICATED\_FULLY"* />

<security:anonymous enabled=*"false"* />

<security:http-basic entry-point-ref=*"oauthAuthenticationEntryPoint"* />

<security:custom-filter ref=*"clientCredentialsTokenEndpointFilter"*

before=*"BASIC\_AUTH\_FILTER"* />

<security:access-denied-handler ref=*"oauthAccessDeniedHandler"* />

</security:http>

<security:http pattern=*"/register/\*\*"*>

<security:intercept-url pattern=*"/register/client"*

access=*"IS\_AUTHENTICATED\_ANONYMOUSLY"* />

<security:form-login authentication-failure-url=*"/login?authentication\_error=true"*

default-target-url=*"/index"* login-page=*"/login"* login-processing-url=*"/login.do"* />

<security:logout logout-success-url=*"/index"*

logout-url=*"/logout.do"* />

<security:anonymous />

</security:http>

<security:http access-denied-page=*"/login?authorization\_error=true"*

disable-url-rewriting=*"true"*>

<security:intercept-url pattern=*"/oauth/\*\*"*

access=*"ROLE\_USER"* />

<security:intercept-url pattern=*"/\*\*"*

access=*"IS\_AUTHENTICATED\_ANONYMOUSLY"* />

<security:form-login authentication-failure-url=*"/login?authentication\_error=true"*

default-target-url=*"/index"* login-page=*"/login"* login-processing-url=*"/login.do"* />

<security:logout logout-success-url=*"/index"*

logout-url=*"/logout.do"* />

<security:anonymous />

</security:http>

<security:authentication-manager id=*"clientAuthenticationManager"*>

<security:authentication-provider

user-service-ref=*"clientDetailsUserService"* />

</security:authentication-manager>

<bean id=*"clientCredentialsTokenEndpointFilter"*

class=*"org.springframework.security.oauth2.provider.client.ClientCredentialsTokenEndpointFilter"*>

<property name=*"authenticationManager"* ref=*"clientAuthenticationManager"* />

</bean>

<bean id=*"clientDetailsUserService"*

class=*"org.springframework.security.oauth2.provider.client.ClientDetailsUserDetailsService"*>

<constructor-arg ref=*"clientDetails"* />

</bean>

<oauth2:authorization-server

client-details-service-ref=*"clientDetails"* token-services-ref=*"tokenServices"*>

<oauth2:authorization-code />

<oauth2:implicit />

<oauth2:refresh-token />

<oauth2:client-credentials />

<oauth2:password />

</oauth2:authorization-server>

<bean id=*"clientDetails"* class=*"com.ddlab.rnd.security.oauth2.ClientDetailsServiceImpl"* />

<bean id=*"oauthAccessDeniedHandler"*

class=*"org.springframework.security.oauth2.provider.error.OAuth2AccessDeniedHandler"* />

<bean id=*"oauthAuthenticationEntryPoint"*

class=*"org.springframework.security.oauth2.provider.error.OAuth2AuthenticationEntryPoint"* />

<bean id=*"accessDecisionManager"* class=*"org.springframework.security.access.vote.UnanimousBased"*

xmlns=*"http://www.springframework.org/schema/beans"*>

<constructor-arg>

<list>

<bean class=*"org.springframework.security.oauth2.provider.vote.ScopeVoter"* />

<bean class=*"org.springframework.security.access.vote.RoleVoter"* />

<bean class=*"org.springframework.security.access.vote.AuthenticatedVoter"* />

</list>

</constructor-arg>

</bean>

<security:authentication-manager alias=*"authenticationManager"*>

<security:authentication-provider>

<security:user-service>

<security:user name=*"a"* password=*"a"* authorities=*"ROLE\_USER"* />

<security:user name=*"b"* password=*"b"* authorities=*"ROLE\_USER"* />

<security:user name=*"d"* password=*"d"* authorities=*"ROLE\_USER"* />

</security:user-service>

</security:authentication-provider>

</security:authentication-manager>

<bean id=*"tokenStore"*

class=*"org.springframework.security.oauth2.provider.token.InMemoryTokenStore"*>

</bean>

<bean id=*"tokenServices"*

class=*"org.springframework.security.oauth2.provider.token.DefaultTokenServices"*>

<property name=*"tokenStore"* ref=*"tokenStore"* />

<property name=*"supportRefreshToken"* value=*"true"* />

<property name=*"accessTokenValiditySeconds"* value=*"3600"* />

<property name=*"refreshTokenValiditySeconds"* value=*"36000"* />

</bean>

<oauth2:expression-handler id=*"oauthExpressionHandler"* />

<oauth2:web-expression-handler id=*"oauthWebExpressionHandler"* />

</beans>

# **Spring Servlet Configuration(WEB-INF/spring-servlet.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.1.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.1.xsd*

*http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-3.1.xsd*

*http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-3.1.xsd"*

xmlns:c=*"http://www.springframework.org/schema/c"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:mvc=*"http://www.springframework.org/schema/mvc"* xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:util=*"http://www.springframework.org/schema/util"*>

<context:component-scan base-package=*"com.ddlab.rnd.security.\*"*/>

<mvc:annotation-driven />

<bean

class=*"org.springframework.web.servlet.view.ContentNegotiatingViewResolver"*>

<property name=*"mediaTypes"*>

<map>

<entry key=*"xml"* value=*"application/xml"* />

<entry key=*"html"* value=*"text/html"* />

<entry key=*"json"* value=*"application/json"* />

</map>

</property>

<property name=*"viewResolvers"*>

<list>

<bean class=*"org.springframework.web.servlet.view.BeanNameViewResolver"* />

<bean

class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>

<property name=*"prefix"* value=*"/WEB-INF/jsp/"* />

<property name=*"suffix"* value=*".jsp"* />

</bean>

</list>

</property>

<property name=*"defaultViews"*>

<list>

<bean

class=*"org.springframework.web.servlet.view.json.MappingJacksonJsonView"* />

</list>

</property>

</bean>

</beans>

# **WEB.xml(WEB-INF/web.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app version=*"2.5"* xmlns=*"http://java.sun.com/xml/ns/javaee"*

xmlns:xml=*"http://www.w3.org/XML/1998/namespace"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd "*>

<welcome-file-list>

<welcome-file>/index</welcome-file>

</welcome-file-list>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>

classpath:spring/security-config.xml

</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.request.RequestContextListener</listener-class>

</listener>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<filter>

<filter-name>springSecurityFilterChain</filter-name>

<filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>

</filter>

<filter-mapping>

<filter-name>springSecurityFilterChain</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<servlet>

<servlet-name>spring</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>spring</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

# **Java Code for Spring Security and for REST Call**

## **AccessConfirmationController.java**

**package** com.ddlab.rnd.security.oauth2;

**import** java.util.Map;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.security.oauth2.provider.AuthorizationRequest;

**import** org.springframework.security.oauth2.provider.ClientDetails;

**import** org.springframework.security.oauth2.provider.ClientDetailsService;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.SessionAttributes;

**import** org.springframework.web.servlet.ModelAndView;

/\*\*

\* Controller for retrieving the model for and displaying the confirmation page for access to a protected resource.

\*

\* **@author** Ryan Heaton

\*/

@Controller

@SessionAttributes("authorizationRequest")

**public** **class** AccessConfirmationController {

**private** ClientDetailsService clientDetailsService;

@RequestMapping("/oauth/confirm\_access")

**public** ModelAndView getAccessConfirmation(Map<String, Object> model) **throws** Exception {

AuthorizationRequest clientAuth = (AuthorizationRequest) model.remove("authorizationRequest");

ClientDetails client = clientDetailsService.loadClientByClientId(clientAuth.getClientId());

model.put("auth\_request", clientAuth);

model.put("client", client);

**return** **new** ModelAndView("access\_confirmation", model);

}

@RequestMapping("/oauth/error")

**public** String handleError(Map<String,Object> model) **throws** Exception {

// We can add more stuff to the model here for JSP rendering. If the client was a machine then

// the JSON will already have been rendered.

model.put("message", "There was a problem with the OAuth2 protocol");

**return** "oauth\_error";

}

@Autowired

**public** **void** setClientDetailsService(ClientDetailsService clientDetailsService) {

**this**.clientDetailsService = clientDetailsService;

}

}

## **ClientDetailsServiceImpl.java**

**package** com.ddlab.rnd.security.oauth2;

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.springframework.security.core.GrantedAuthority;

**import** org.springframework.security.core.authority.SimpleGrantedAuthority;

**import** org.springframework.security.oauth2.common.exceptions.OAuth2Exception;

**import** org.springframework.security.oauth2.provider.BaseClientDetails;

**import** org.springframework.security.oauth2.provider.ClientDetails;

**import** org.springframework.security.oauth2.provider.ClientDetailsService;

**import** org.springframework.security.oauth2.provider.NoSuchClientException;

**public** **class** ClientDetailsServiceImpl **implements** ClientDetailsService {

/\*

\* From this method, get the user details

\* This class is tailor made, manually doing everything for

\* better customization

\*/

@Override

**public** ClientDetails loadClientByClientId(String clientId) **throws** OAuth2Exception {

System.***out***.println("-----------------ClientDetailsServiceImpl--------------------------");

**if** (clientId.equals("client1")) {

List<String> authorizedGrantTypes = **new** ArrayList<>();

authorizedGrantTypes.add("password");

authorizedGrantTypes.add("refresh\_token");

authorizedGrantTypes.add("client\_credentials");

authorizedGrantTypes.add("authorization\_code");//IMP

authorizedGrantTypes.add("implicit");//IMP

BaseClientDetails clientDetails = **new** BaseClientDetails();

clientDetails.setClientId("client1");

clientDetails.setClientSecret("client1");

clientDetails.setAuthorizedGrantTypes(authorizedGrantTypes);

GrantedAuthority ga = **new** SimpleGrantedAuthority("ROLE\_USER");

List<GrantedAuthority> gaList = **new** ArrayList<GrantedAuthority>();

gaList.add(ga);

clientDetails.setAuthorities(gaList);

List<String> scopeList = **new** ArrayList<String>();

scopeList.add("read");

scopeList.add("write");

scopeList.add("delete");

clientDetails.setScope(scopeList);

**return** clientDetails;

} **else** **if** (clientId.equals("client2")) {

List<String> authorizedGrantTypes = **new** ArrayList<>();

authorizedGrantTypes.add("password");

authorizedGrantTypes.add("refresh\_token");

authorizedGrantTypes.add("client\_credentials");

BaseClientDetails clientDetails = **new** BaseClientDetails();

clientDetails.setClientId("client2");

clientDetails.setClientSecret("client2");

clientDetails.setAuthorizedGrantTypes(authorizedGrantTypes);

**return** clientDetails;

}

**else** {

**throw** **new** NoSuchClientException("No client with requested id: "

+ clientId);

}

}

}

## **LoginController.java**

**package** com.ddlab.rnd.security.oauth2;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.RequestMapping;

@Controller

**public** **class** LoginController {

@RequestMapping("login")

**public** String login() {

**return** "login";

}

@RequestMapping("logout")

**public** String logout() {

**return** "logout";

}

@RequestMapping("index")

**public** String index() {

**return** "index";

}

@RequestMapping("request\_token\_authorized")

**public** String request\_token\_authorized() {

**return** "request\_token\_authorized";

}

}

## **ProtectedContoller.java**

**package** com.ddlab.rnd.security.oauth2;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.bind.annotation.ResponseBody;

@Controller

@RequestMapping("/")

**public** **class** ProtectedController {

@RequestMapping(value = "/test/test/{id}", method = RequestMethod.***GET***)

@ResponseBody

**public** String tokenManagment( @PathVariable("id") String id ) {

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ID\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*:::::"+id);

**return** "You got protected information from server";

}

}

# **JSP Files (WEB-INF/jsp)**

## **Login.jsp**

<%@ taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>

<%@ taglib uri=*"http://www.springframework.org/tags"* prefix=*"spring"*%>

<!DOCTYPE html>

<html>

<head>

<title>OAUTH2 Authentication in Spring Security</title>

</head>

<body>

<h1>InMemory based OAUTH2 Authentication in ZARA Security</h1>

<div id=*"content"*>

<!-- For displaying errors -->

<c:if test=*"*${**not empty** param.authentication\_error}*"*>

<h1>Woops!</h1>

<p class=*"error"*>Your login attempt was not successful.</p>

</c:if>

<c:if test=*"*${**not empty** param.authorization\_error}*"*>

<h1>Woops!</h1>

<p class=*"error"*>You are not permitted to access that resource.</p>

</c:if>

<!-- For displaying errors -->

<h2>Login with your credentials</h2>

<h3>Username : d, password : d</h3>

<form id=*"loginForm"* name=*"loginForm"*

action=*"*<c:url value=*"/login.do"*/>*"* method=*"post"*>

<table>

<tr>

<td><strong>Username:</strong></td>

<td><input type=*'text'* name=*'j\_username'* value=*"d"*></td>

</tr>

<tr>

<td><strong>Password:</strong></td>

<td><input type=*"password"* name=*'j\_password'* value=*"d"*></td>

</tr>

<tr>

<td/>

<td><input name=*"login"* value=*"Login"* type=*"submit"* /></td>

</tr>

</table>

</form>

</div>

</body>

</html>

## **Access\_confirmation.jsp**

<%@ page

import=*"org.springframework.security.core.AuthenticationException"*%>

<%@ page

import=*"org.springframework.security.web.authentication.AbstractAuthenticationProcessingFilter"*%>

<%@ page

import=*"org.springframework.security.oauth2.common.exceptions.UnapprovedClientAuthenticationException"*%>

<%@ taglib prefix=*"authz"*

uri=*"http://www.springframework.org/security/tags"*%>

<%@ taglib uri=*"http://www.springframework.org/tags"* prefix=*"spring"*%>

<%@ taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"*%>

<!DOCTYPE html>

<html>

<head>

<title>Access Confirmation</title>

</head>

<body>

<h1>Access Confirmation</h1>

<div id=*"content"*>

<%

**if** (session

.getAttribute(AbstractAuthenticationProcessingFilter.SPRING\_SECURITY\_LAST\_EXCEPTION\_KEY) != **null**

&& !(session

.getAttribute(AbstractAuthenticationProcessingFilter.SPRING\_SECURITY\_LAST\_EXCEPTION\_KEY) **instanceof** UnapprovedClientAuthenticationException)) {

%>

<div class=*"error"*>

<h2>Woops!</h2>

<p>

Access could not be granted. (<%=((AuthenticationException) session

.getAttribute(AbstractAuthenticationProcessingFilter.SPRING\_SECURITY\_LAST\_EXCEPTION\_KEY))

.getMessage()%>)

</p>

</div>

<%

}

%>

<c:remove scope=*"session"* var=*"SPRING\_SECURITY\_LAST\_EXCEPTION"* />

<authz:authorize ifAllGranted=*"ROLE\_USER"*>

<h2>Please Confirm</h2>

<p>

You hereby authorize "

<c:out value=*"*${client.clientId}*"* />

" to access your ZARA protected resources with following scopes :

</p>

<ul>

<c:forEach var=*"scope"* items=*"*${auth\_request.scope}*"*>

<li>${scope}</li>

</c:forEach>

</ul>

<table>

<tr>

<td>

<form id=*"confirmationForm"* name=*"confirmationForm"*

action=*"*<%=request.getContextPath()%>*/oauth/authorize"*

method=*"post"*>

<input name=*"user\_oauth\_approval"* value=*"true"* type=*"hidden"* />

<label><input name=*"authorize"* value=*"Authorize"*

type=*"submit"*></label>

</form>

</td>

<td>

<form id=*"denialForm"* name=*"denialForm"*

action=*"*<%=request.getContextPath()%>*/oauth/authorize"*

method=*"post"*>

<input name=*"user\_oauth\_approval"* value=*"false"* type=*"hidden"* />

<label><input name=*"deny"* value=*"Deny"* type=*"submit"*></label>

</form>

</td>

</tr>

</table>

</authz:authorize>

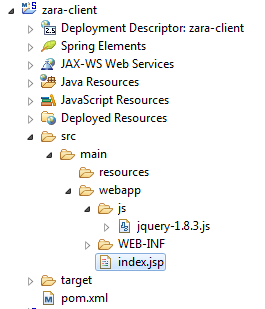
</div>

</body>

</html>

# **Client Application**

This is another application having the following structure. It has nothing except the jsp page to access the authorized server information. The structure is given below.



# **JSP page**

## **Index.jsp page**

<!DOCTYPE html>

<html>

<head>

<script type=*"text/javascript"* src=*"js/jquery-1.8.3.js"*></script>

<script type=*"text/javascript"*>

**var** authServerUrl = "http://localhost:8080/zara-auth";

**var** clientUrl = "http://localhost:8080/zara-client/index.jsp";

**function** Redirect() {

**var** clientId = "client1"; /\*Fixed for an application\*/

**var** scope = "read write delete";

**var** responseType = "token";

**var** serverUrl = authServerUrl+"/oauth/authorize";

serverUrl = serverUrl + "?response\_type=" + responseType

+ "&client\_id=" + clientId + "&scope=" + scope;

**var** authUrl = serverUrl + "&redirect\_uri=" + clientUrl;

$(location).attr('href', authUrl);

}

**function** check() {

**try** {

**var** currentUrl = window.location.href;

**var** access\_token = currentUrl.match(/\#(?:access\_token)\=([\S\s]\*?)\&/)[1];

**var** serverURL = authServerUrl+"/test/test/123";

$.ajax({

type : "GET",

url : serverURL,

dataType : 'text',

async : **true**,

headers : {

"Authorization" : "bearer " + access\_token

},

success : **function**(response) {

//alert(response);

$('#resultId').html(response).css({"color":"green"});

},

error : **function**(result) {

$('#resultId').html(" Unexpected Server Error ").css({"color":"red"});

}

});

}

**catch**(err) {

**var** message = "Unexpected error, probably you have to login by clicking above the clink.";

$('#resultId').html(message).css({"color":"red"});;

}

}

</script>

</head>

<body>

<h1>Click below the link to login before submitting zara cluster id.</h1>

<h2>

<a id=*"loginLink"* href=*"#"* onclick="Redirect();" style="text-decoration: *none*">

<strong>Click here to login</strong>

</a>

</h2>

<table>

<tr>

<td>Enter ZARA cluster id:</td>

<td><input type=*"text"* name=*"textId"* value=*""* id=*"studentId"*></td>

<td><input type=*"button"* value=*"Submit"* onclick="check();"></td>

</tr>

</table>

<br>

<div id=*"responseId"*>

<label id=*"resultId"*> </label>

<label id=*"errorId"* style="color: *red*"></label>

</div>

<input type=*"hidden"* id=*"tokenId"* value=*""* />

</body>

</html>

# **How to Use**

InMemory OAUTH2 authentication in Spring security without using Database

========================================================================

To Access in browser

http://localhost:8080/zara-client/

To access in REST Client

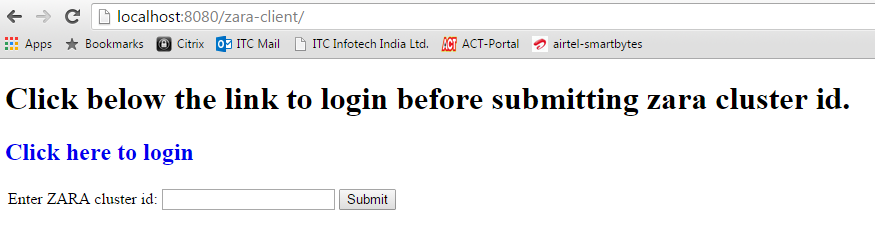
http://localhost:8080/zara-auth/test/test/11

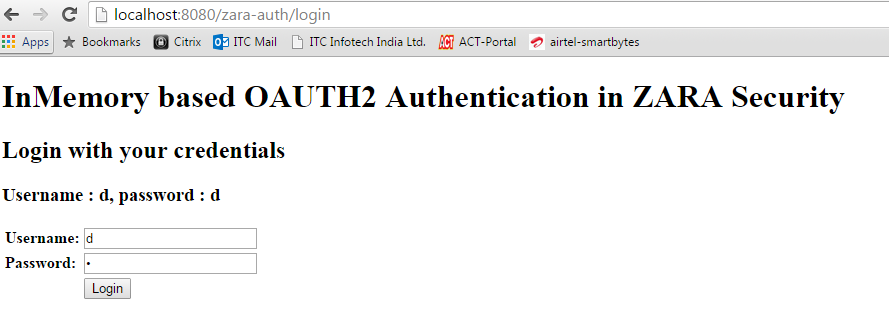
Pass the following as header

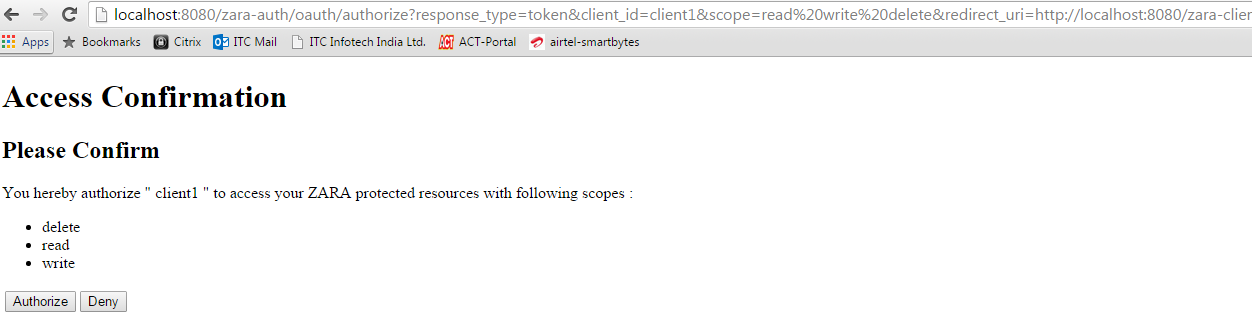
Authorization

bearer 8f0e4006-2e8d-424e-8de0-5e4e3008f17c

Complete application flow is given below.









OAUTH2 with 3 Legged Authentication – JDBC Based

In this everything is same as described above, the only difference is the **spring-config.xml** file.

# **Resources/spring/security-config.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:aop=*"http://www.springframework.org/schema/aop"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"* xmlns:oauth2=*"http://www.springframework.org/schema/security/oauth2"*

xmlns:p=*"http://www.springframework.org/schema/p"* xmlns:security=*"http://www.springframework.org/schema/security"*

xmlns:tx=*"http://www.springframework.org/schema/tx"* xmlns:mvc=*"http://www.springframework.org/schema/mvc"*

xsi:schemaLocation=*"http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-3.1.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.1.xsd*

*http://www.springframework.org/schema/security/oauth2 http://www.springframework.org/schema/security/spring-security-oauth2.xsd*

*http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd*

*http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security-3.1.xsd*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.1.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.1.xsd"*>

<context:component-scan base-package=*"com.ddlab.rnd.security.\*"* />

<mvc:default-servlet-handler />

<bean id=*"dataSource"*

class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"* />

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/test"* />

<property name=*"username"* value=*"deba"* />

<property name=*"password"* value=*"deba"* />

</bean>

<security:http pattern=*"/test/\*\*"* create-session=*"never"*

entry-point-ref=*"oauthAuthenticationEntryPoint"*

access-decision-manager-ref=*"accessDecisionManager"*>

<security:anonymous enabled=*"false"* />

<security:intercept-url pattern=*"/test/test/\*\*"*

access=*"ROLE\_USER"* />

<security:custom-filter ref=*"resourceServerFilter"*

before=*"PRE\_AUTH\_FILTER"* />

<security:access-denied-handler ref=*"oauthAccessDeniedHandler"* />

</security:http>

<oauth2:resource-server id=*"resourceServerFilter"*

token-services-ref=*"tokenServices"* />

<security:http pattern=*"/oauth/token"* create-session=*"stateless"*

authentication-manager-ref=*"clientAuthenticationManager"*

entry-point-ref=*"oauthAuthenticationEntryPoint"*>

<security:intercept-url pattern=*"/oauth/token"*

access=*"IS\_AUTHENTICATED\_FULLY"* />

<security:anonymous enabled=*"false"* />

<security:http-basic entry-point-ref=*"oauthAuthenticationEntryPoint"* />

<security:custom-filter ref=*"clientCredentialsTokenEndpointFilter"*

before=*"BASIC\_AUTH\_FILTER"* />

<security:access-denied-handler ref=*"oauthAccessDeniedHandler"* />

</security:http>

<security:http pattern=*"/register/\*\*"*>

<security:intercept-url pattern=*"/register/client"*

access=*"IS\_AUTHENTICATED\_ANONYMOUSLY"* />

<security:form-login authentication-failure-url=*"/login?authentication\_error=true"*

default-target-url=*"/index"* login-page=*"/login"* login-processing-url=*"/login.do"* />

<security:logout logout-success-url=*"/index"*

logout-url=*"/logout.do"* />

<security:anonymous />

</security:http>

<security:http access-denied-page=*"/login?authorization\_error=true"*

disable-url-rewriting=*"true"*>

<security:intercept-url pattern=*"/oauth/\*\*"*

access=*"ROLE\_USER"* />

<security:intercept-url pattern=*"/\*\*"*

access=*"IS\_AUTHENTICATED\_ANONYMOUSLY"* />

<security:form-login authentication-failure-url=*"/login?authentication\_error=true"*

default-target-url=*"/index"* login-page=*"/login"* login-processing-url=*"/login.do"* />

<security:logout logout-success-url=*"/index"*

logout-url=*"/logout.do"* />

<security:anonymous />

<security:remember-me data-source-ref=*"dataSource"*

user-service-ref=*"jdbcUserService"* />

</security:http>

<security:authentication-manager id=*"clientAuthenticationManager"*>

<security:authentication-provider

user-service-ref=*"clientDetailsUserService"* />

</security:authentication-manager>

<bean id=*"clientCredentialsTokenEndpointFilter"*

class=*"org.springframework.security.oauth2.provider.client.ClientCredentialsTokenEndpointFilter"*>

<property name=*"authenticationManager"* ref=*"clientAuthenticationManager"* />

</bean>

<bean id=*"clientDetailsUserService"*

class=*"org.springframework.security.oauth2.provider.client.ClientDetailsUserDetailsService"*>

<constructor-arg ref=*"clientDetails"* />

</bean>

<oauth2:authorization-server

client-details-service-ref=*"clientDetails"* token-services-ref=*"tokenServices"*>

<oauth2:authorization-code />

<oauth2:implicit />

<oauth2:refresh-token />

<oauth2:client-credentials />

<oauth2:password />

</oauth2:authorization-server>

<bean id=*"clientDetails"*

class=*"org.springframework.security.oauth2.provider.JdbcClientDetailsService"*>

<constructor-arg name=*"dataSource"* ref=*"dataSource"*></constructor-arg>

</bean>

<bean id=*"oauthAccessDeniedHandler"*

class=*"org.springframework.security.oauth2.provider.error.OAuth2AccessDeniedHandler"* />

<bean id=*"oauthAuthenticationEntryPoint"*

class=*"org.springframework.security.oauth2.provider.error.OAuth2AuthenticationEntryPoint"* />

<bean id=*"accessDecisionManager"* class=*"org.springframework.security.access.vote.UnanimousBased"*

xmlns=*"http://www.springframework.org/schema/beans"*>

<constructor-arg>

<list>

<bean class=*"org.springframework.security.oauth2.provider.vote.ScopeVoter"* />

<bean class=*"org.springframework.security.access.vote.RoleVoter"* />

<bean class=*"org.springframework.security.access.vote.AuthenticatedVoter"* />

</list>

</constructor-arg>

</bean>

<!-- Server side configuration -->

<security:authentication-manager alias=*"authenticationManager"*>

<security:authentication-provider>

<!-- <security:password-encoder hash="md5" /> -->

<security:jdbc-user-service id=*"jdbcUserService"*

data-source-ref=*"dataSource"* />

</security:authentication-provider>

</security:authentication-manager>

<bean id=*"tokenStore"*

class=*"org.springframework.security.oauth2.provider.token.JdbcTokenStore"*>

<constructor-arg name=*"dataSource"* ref=*"dataSource"*></constructor-arg>

</bean>

<bean id=*"tokenServices"*

class=*"org.springframework.security.oauth2.provider.token.DefaultTokenServices"*>

<property name=*"tokenStore"* ref=*"tokenStore"* />

<property name=*"supportRefreshToken"* value=*"true"* />

<property name=*"accessTokenValiditySeconds"* value=*"3600"* />

<property name=*"refreshTokenValiditySeconds"* value=*"36000"* />

</bean>

<oauth2:expression-handler id=*"oauthExpressionHandler"* />

<oauth2:web-expression-handler id=*"oauthWebExpressionHandler"* />

</beans>

# **Database Tables are given below**

/\* Drop Tables related to OAUTH2 \*/

**drop** **table** if **exists** oauth\_client\_details;

**drop** **table** if **exists** oauth\_access\_token;

**drop** **table** if **exists** oauth\_refresh\_token;

**drop** **table** if **exists** oauth\_client\_token;

**drop** **table** if **exists** oauth\_code;

/\*Drop Tables related users and granted authorities\*/

**drop** **table** if **exists** authorities;

**drop** **table** if **exists** users;

/\* Create Tables related users and granted authorities\*/

**create** **table** if **not** **exists** users(

username **varchar**(50) **not** **null** **primary** **key**,

password **varchar**(50) **not** **null**,

enabled **boolean** **not** **null**

);

**create** **table** if **not** **exists** authorities (

username **varchar**(50) **not** **null**,

authority **varchar**(50) **not** **null**,

**constraint** fk\_authorities\_users **foreign** **key**(username) **references** users(username),

**unique** **index** ix\_auth\_username (username,authority)

);

/\* Create Tables related to OAUTH2 \*/

**create** **table** if **not** **exists** oauth\_client\_details (

client\_id **VARCHAR**(128) **PRIMARY** **KEY**,

resource\_ids **VARCHAR**(128),

client\_secret **VARCHAR**(128),

scope **VARCHAR**(128),

authorized\_grant\_types **VARCHAR**(128),

web\_server\_redirect\_uri **VARCHAR**(128),

authorities **VARCHAR**(128),

access\_token\_validity **INTEGER**,

refresh\_token\_validity **INTEGER**,

additional\_information **VARCHAR**(128)

);

**create** **table** if **not** **exists** oauth\_access\_token (

token\_id **VARCHAR**(256),

token longblob,

authentication\_id **VARCHAR**(256),

user\_name **VARCHAR**(256),

client\_id **VARCHAR**(256),

authentication longblob,

refresh\_token **VARCHAR**(256)

);

**create** **table** if **not** **exists** oauth\_refresh\_token (

token\_id **VARCHAR**(256),

token longblob,

authentication longblob

);

**create** **table** if **not** **exists** oauth\_client\_token (

token\_id **VARCHAR**(256),

token longblob,

authentication\_id **VARCHAR**(256),

user\_name **VARCHAR**(256),

client\_id **VARCHAR**(256)

);

**create** **table** if **not** **exists** oauth\_code (

code **VARCHAR**(256), authentication longblob

);

/\* Sample Insertion Scripts \*/

**insert** **into** oauth\_client\_details **values** (

'client1',

**null**,

**null**,

'read,write,delete',

'password,authorization\_code,refresh\_token,implicit',

**null**,

'ROLE\_CLIENT, ROLE\_TRUSTED\_CLIENT',

**null**,

**null**,

**null**

);

**insert** **into** users **values** ('a','a',1);

**insert** **into** users **values** ('b','b',1);

**insert** **into** users **values** ('c','c',1);

**insert** **into** users **values** ('d','d',1);

**insert** **into** authorities **values** ('a','ROLE\_USER');

**insert** **into** authorities **values** ('a','ROLE\_CLIENT');

**insert** **into** authorities **values** ('b','ROLE\_USER') ;

**insert** **into** authorities **values** ('b','ROLE\_CLIENT');

**insert** **into** authorities **values** ('c','ROLE\_ADMIN');

**insert** **into** authorities **values** ('c','ROLE\_TRUSTED\_CLIENT');

**insert** **into** authorities **values** ('d','ROLE\_USER');

**insert** **into** authorities **values** ('d','ROLE\_CLIENT');

**commit**;