Security using Jersey 2.x

a. 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/maven-v4_0_0.xsd">

  <modelVersion>4.0.0</modelVersion>

  <groupId>org.sathyatech</groupId>
  <artifactId>JerseySecureApp</artifactId>
  <packaging>war</packaging>
  <version>1.0</version>
  <name>JerseySecureApp</name>

  <build>
    <finalName>JerseySecureApp</finalName>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>
        <inherited>true</inherited>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <plugin>
                        <artifactId>maven-war-plugin</artifactId>
                        <version>2.4</version>
                        <configuration>
                              <failOnMissingWebXml>false</failOnMissingWebXml>
                        </configuration>
```

```
                    </plugin>
    </plugins>
  </build>



  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.glassfish.jersey</groupId>
        <artifactId>jersey-bom</artifactId>
        <version>${jersey.version}</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
    </dependencies>
  </dependencyManagement>


  <dependencies>
    <dependency>
      <groupId>org.glassfish.jersey.containers</groupId>
      <artifactId>jersey-container-servlet</artifactId>
      <!-- use the following artifactId if you don't need servlet 2.x compatibility -->
      <!-- artifactId>jersey-container-servlet</artifactId -->
    </dependency>
    <dependency>
      <groupId>org.glassfish.jersey.inject</groupId>
      <artifactId>jersey-hk2</artifactId>
    </dependency>
    <!-- uncomment this to get JSON support-->
    <dependency>
      <groupId>org.glassfish.jersey.media</groupId>
      <artifactId>jersey-media-json-binding</artifactId>
    </dependency>
```

```
  </dependencies>
  <properties>
    <jersey.version>2.28</jersey.version>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>
</project>
```

b. AppConfig.java:-

```
package org.sathyatech.app;

import javax.ws.rs.ApplicationPath;

import org.glassfish.jersey.server.ResourceConfig;

/***
 * This file is equals to web.xml
 * Here Provide URL(Path) and
 * packages where our classes exist
 */
@ApplicationPath("/rest")//URL Pattern
public class AppConfig extends ResourceConfig{

        public AppConfig() {
            //write classes under this package only
            packages("org.sathyatech.app");
            //Link Filter with FC
            register(new SecurityFilter());
        }
}
```

c. ProductService.java:-

```
package org.sathyatech.app;

import javax.annotation.security.DenyAll;
import javax.annotation.security.PermitAll;
import javax.annotation.security.RolesAllowed;
import javax.ws.rs.GET;
import javax.ws.rs.Path;

/**
 * This is service Provider class
 *
 */
@Path("/product")
public class ProductService {
	@GET
	@Path("/all")
	@PermitAll
	public String showCode() {
		return "CODE-AB";
	}

	@GET
	@Path("/none")
	@DenyAll
	public String showMode() {
		return "MODE-NONE";
	}

	@GET
	@Path("/few")
	@RolesAllowed({"ADMIN","EMPLOYEE"})
	public String showDetails() {
		return "DETAILS";
	}
}
```

d. SecurityFilter.java:-

```
package org.sathyatech.app;

import java.io.IOException;
import java.lang.reflect.Method;
import java.util.Arrays;
import java.util.List;
import java.util.StringTokenizer;

import javax.annotation.security.DenyAll;
import javax.annotation.security.PermitAll;
import javax.annotation.security.RolesAllowed;
import javax.ws.rs.container.ContainerRequestContext;
import javax.ws.rs.container.ContainerRequestFilter;
import javax.ws.rs.container.ResourceInfo;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.HttpHeaders;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.glassfish.jersey.internal.util.Base64;

public class SecurityFilter
implements ContainerRequestFilter{

        /*** Details of Service class and methods*/
        @Context
        private ResourceInfo resource;

        /**To read Header Parameters from request**/
        @Context
        private HttpHeaders headers;
```

```
	public void filter(ContainerRequestContext req) throws IOException {
		//called method
		Method m=resource.getResourceMethod();
		if(!m.isAnnotationPresent(PermitAll.class)) {
			if(m.isAnnotationPresent(DenyAll.class)) {
				/*** if method has DenyAll annotation abort process**/
				req.abortWith(
						Response
						.ok("NO ACCESS PROVIDED")
						.status(Status.FORBIDDEN)
						.build());
				return;
			}
		}
		//do security check -un,pwd,role
		if(m.isAnnotationPresent(RolesAllowed.class)) {

			//empty or null un/pwd
			List<String> auth=headers.getRequestHeader("Authorization");
			if(auth==null || auth.isEmpty()) {
				req.abortWith(Response.ok("EMPTY DETAILS
FOUND").status(Status.BAD_REQUEST).build());
				return;
			}
			List<String> users=getUserandPwd(auth.get(0));
			List<String>
roles=Arrays.asList(m.getAnnotation(RolesAllowed.class).value());


			//verify user with DB
			if(!isValidUser(users, roles)) {
				//if invaid user then stop process
				req.abortWith(Response.ok("Invalid User
found").status(Status.UNAUTHORIZED).build());
				return;
			}
```

```
		} //roles allowed end

	}//filter end

	/***
	 * This method is used to read un,pwd
	 * as List from Authorization Header
	 * using Base64,Tokenize concept
	 */
	public List<String> getUserandPwd(String auth) {
		//remove basic space
		auth=auth.replaceAll("Basic ", "");
		//decode
		auth=Base64.decodeAsString(auth.getBytes());
		//tokenize
		StringTokenizer str=new StringTokenizer(auth, ":");
		return Arrays.asList(
				str.nextToken(),//un
				str.nextToken() //pwd
				);

	}

	private boolean isValidUser(List<String> users,List<String> roles) {
		if("sam".equals(users.get(0)) && "sam".equals(users.get(1))
				&& roles.contains("ADMIN")) {
			return true;
		}else
			if("khan".equals(users.get(0)) && "khan".equals(users.get(1))
					&& roles.contains("EMPLOYEE")) {
				return true;
			}
		return false;
	}

}// class end
```

FB: https://www.facebook.com/groups/thejavatemple

Email : javabyraghu@gmail.com