3.2 OAuth, SSL, and Log4j



This section will guide you to understand:

* The problem statement on OAuth, SSL, and Log4j using REST Assured
* The solution for the problem statement

**Development Environment:**

* Eclipse IDE
* Java 1.8

This guide has three subsections, namely:

3.2.1 Problem statement for OAuth, SSL, and Log4j

3.2.2 Solution for the problem statement

3.2.3 Pushing the code to GitHub repositories

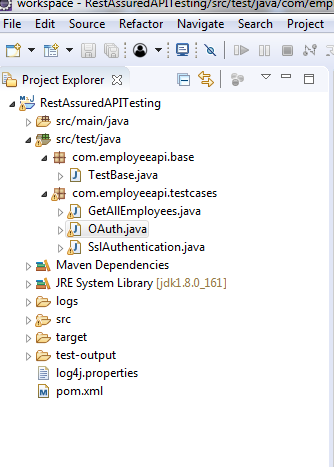
**Step 3.2.1:** Problem statement for OAuth, SSL, and Log4j

* **Objective**: Perform OAuth, SSL authentication for a given REST API, and track the execution using Log4j.
* **Steps involved**:

1. Create a Maven project.
2. Create class to write test cases.
3. Create a test method to handle SSLPeerUnverifiedException and to perform OAuth Authentication.
4. Include logger in the method.

**Step 3.2.2:** Solution for the problem statement

* The project structure looks like this:



* Open Eclipse.
* Click on file---> click on New--->Project.
* Select the Maven project and click on Next.
* Enter the Group id and Artifact id and click on Finish.
* Add the required dependencies to the pom.xml.
* Right click on Project --> New --> File.
* Name the file as “log4j.properties” and click on Finish.
* Open log4j.properties.
* Write the code shown below:

# Root Logger option

log4j.rootLogger=INFO, file, stdout

# Direct log messages to stdout

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.Target=System.out

log4j.appender.stdout.layout=

org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=

%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L %m%n

* Right click on project---> src/test/java---> Package.
* Enter the package name(Ex: com.employeeapi.testcases) and click on Finish.
* Right click on Package---> New---> Class.
* Enter the class name(Ex: OAuth) and click on Finish.
* Write a below code inside OAuth Class:

1. EmployeesRestAPI is the name given to logger.
2. Log4j.properties is the name of the file we created.
3. relaxedHTTPSValidation() is used to handle SSLPeerUnverifiedException.
4. “a2c46473d65826bb118e5ae7e260d4cf604c8e982” is the key.

**package** com.employeeapi.testcases;

**import** org.apache.log4j.Level;

**import** org.apache.log4j.Logger;

**import** org.apache.log4j.PropertyConfigurator;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**import** **static** io.restassured.RestAssured.**\***;

**public** **class** OAuth{

**public** **Logger** logger;

**@BeforeClass**

**public** void setup()

{

logger=**Logger**.getLogger("EmployeesRestAPI");

PropertyConfigurator.configure("Log4j.properties");

logger.setLevel(**Level**.DEBUG);

}

**@Test**

**public** void Oauth()

{

logger.info("\*\*\*\*\*\*\*\*\*\*\*statrt of OAuth

and SSLPeerUnverifiedException handling\*\*\*\*\*\*\*\*\*\*");

given().relaxedHTTPSValidation()

.auth()

.oauth2("a2c46473d65826bb118e5ae7e260d4cf604c8e982")

.post("http://192.168.1.207:8080/api/employee/search

/1597534560")

.then().statusCode(200);

logger.info("\*\*\*\*\*\*\*\*\*\*\*End of OAuth

and SSLPeerUnverifiedException handling\*\*\*\*\*\*\*\*\*\*");

}

}

* Right click on OAuth class --> Run As --> TestNG Test and verify the output in the console:

