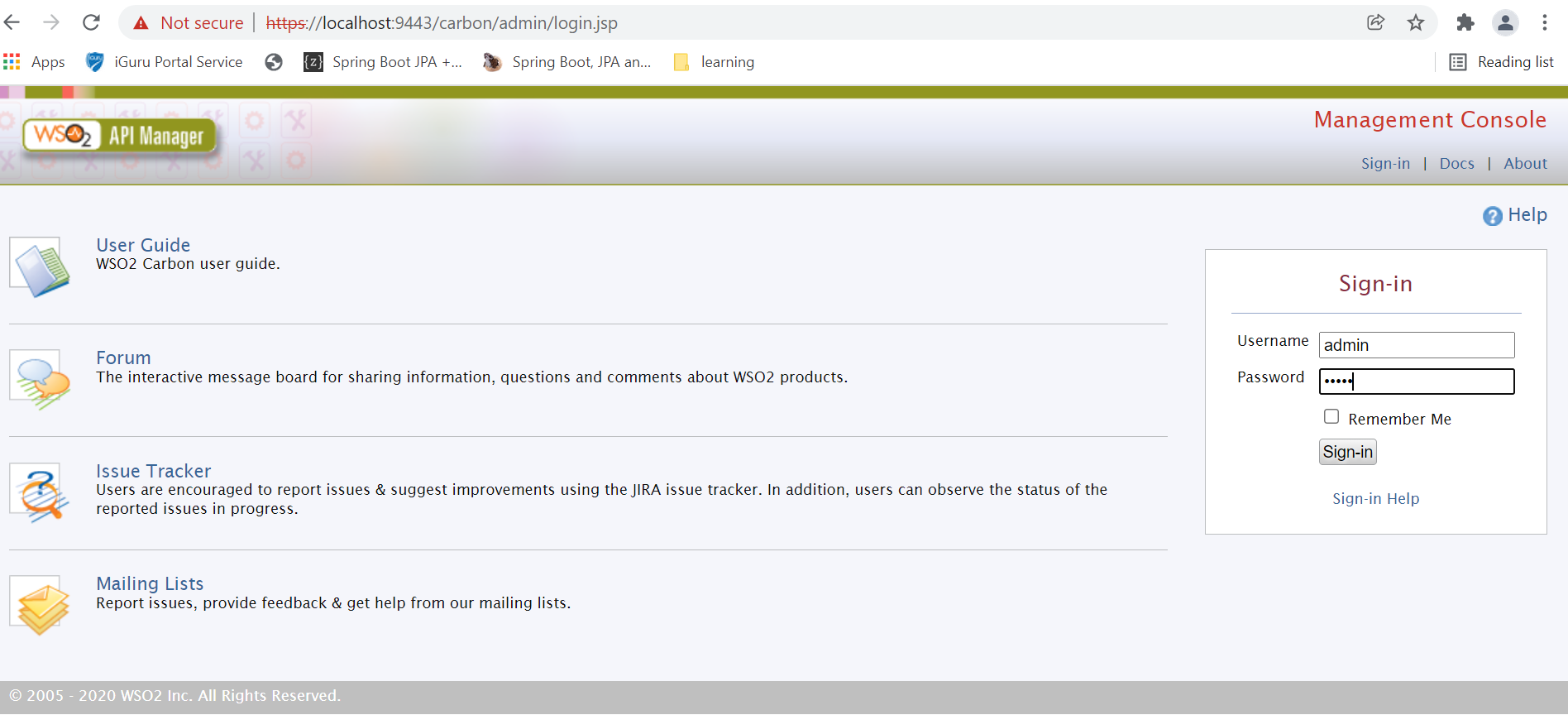
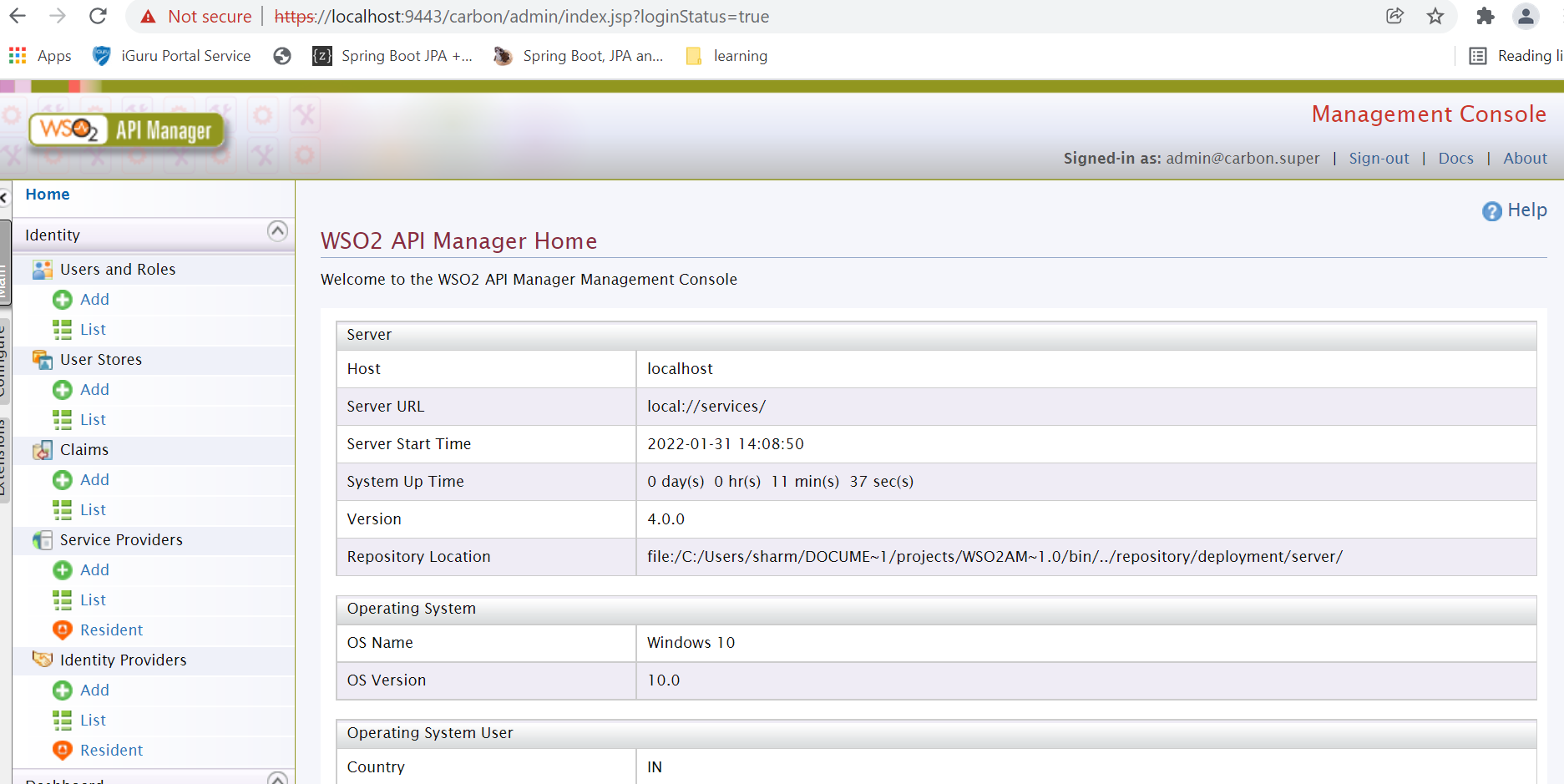
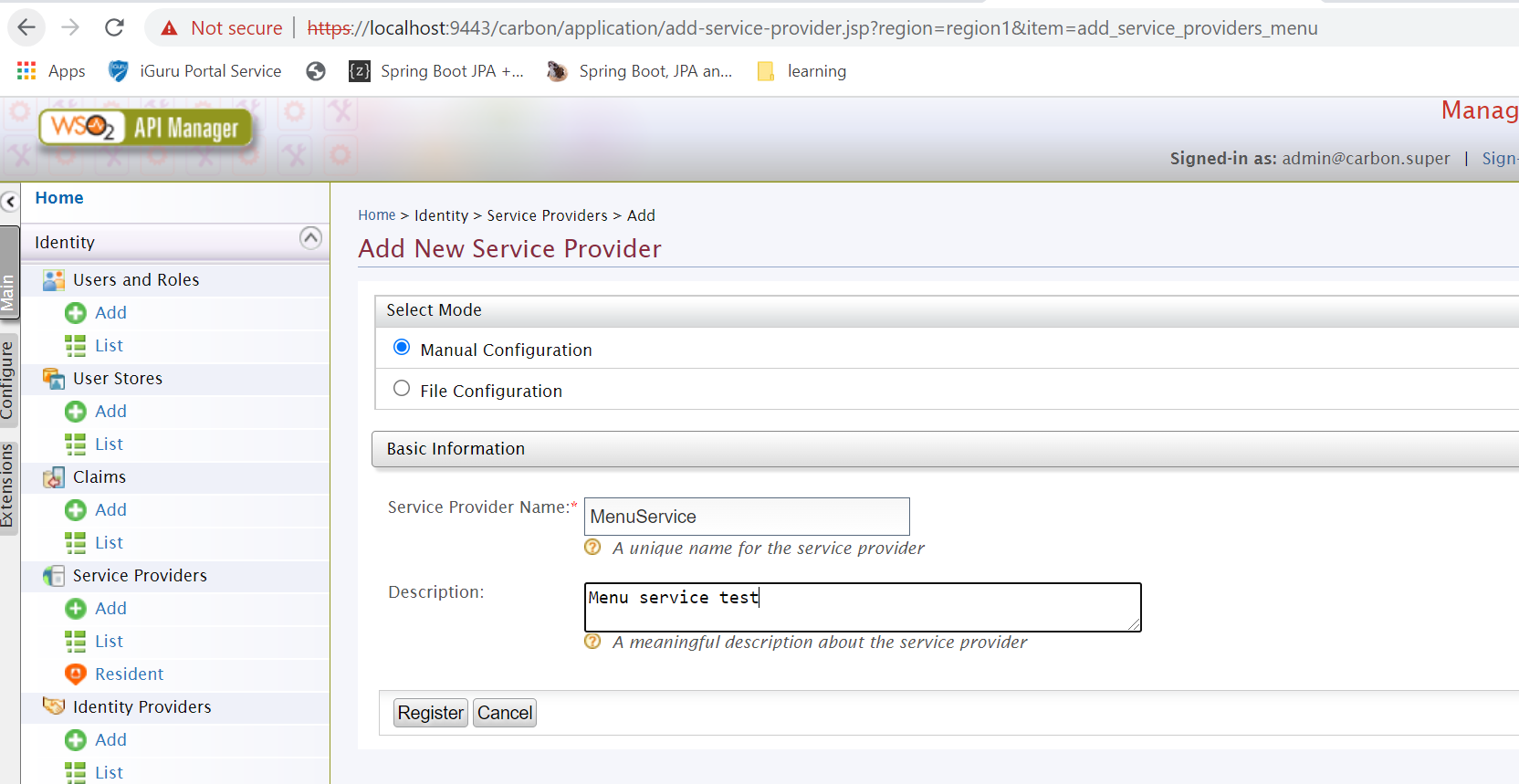
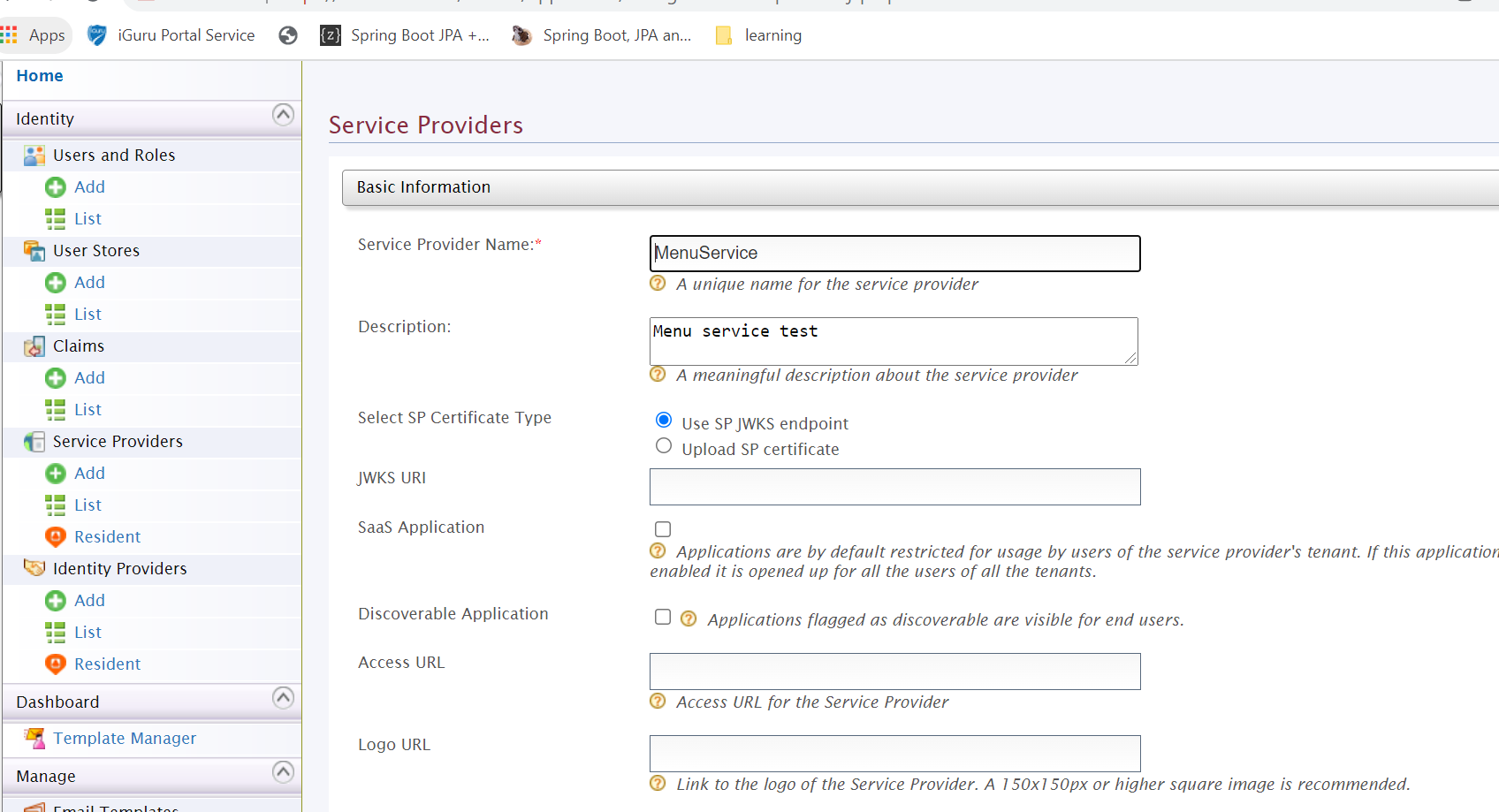
oauth2-login-for-spring-boot-application-with-wso2-identity-server (Without SSL certificate)

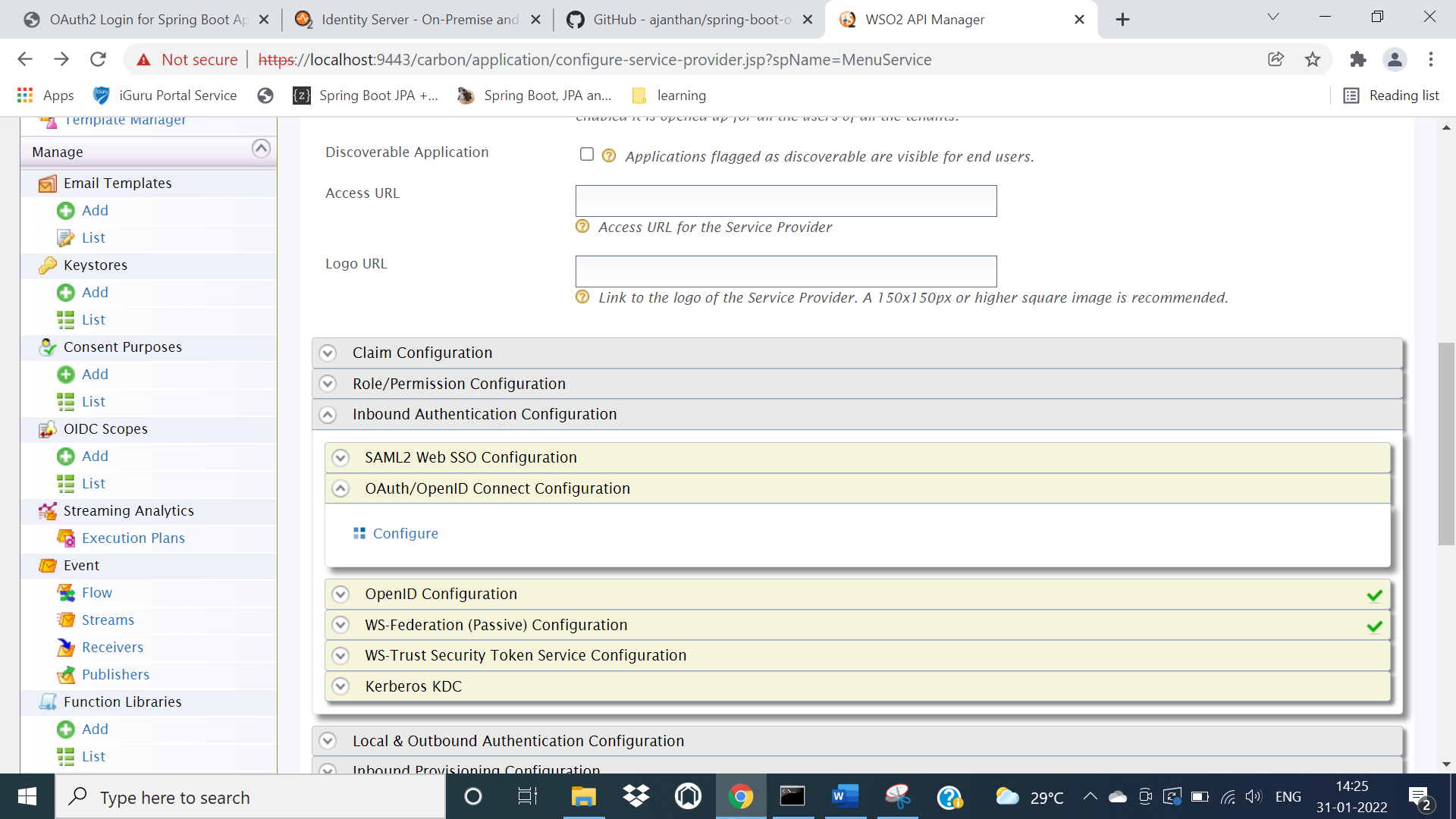
Login on <https://localhost:9443/carbon> and login using default credentials(admin:admin).

Add a service provider by clicking home>Service Providers>Add as shown in the screenshot.

Select the Manual configuration and give a meaning full name to the service provider and click register. 

Then it will open details page of your Service Providers like below

Click on inbound authentication >OAuth2/OpenID Connect>Configure .

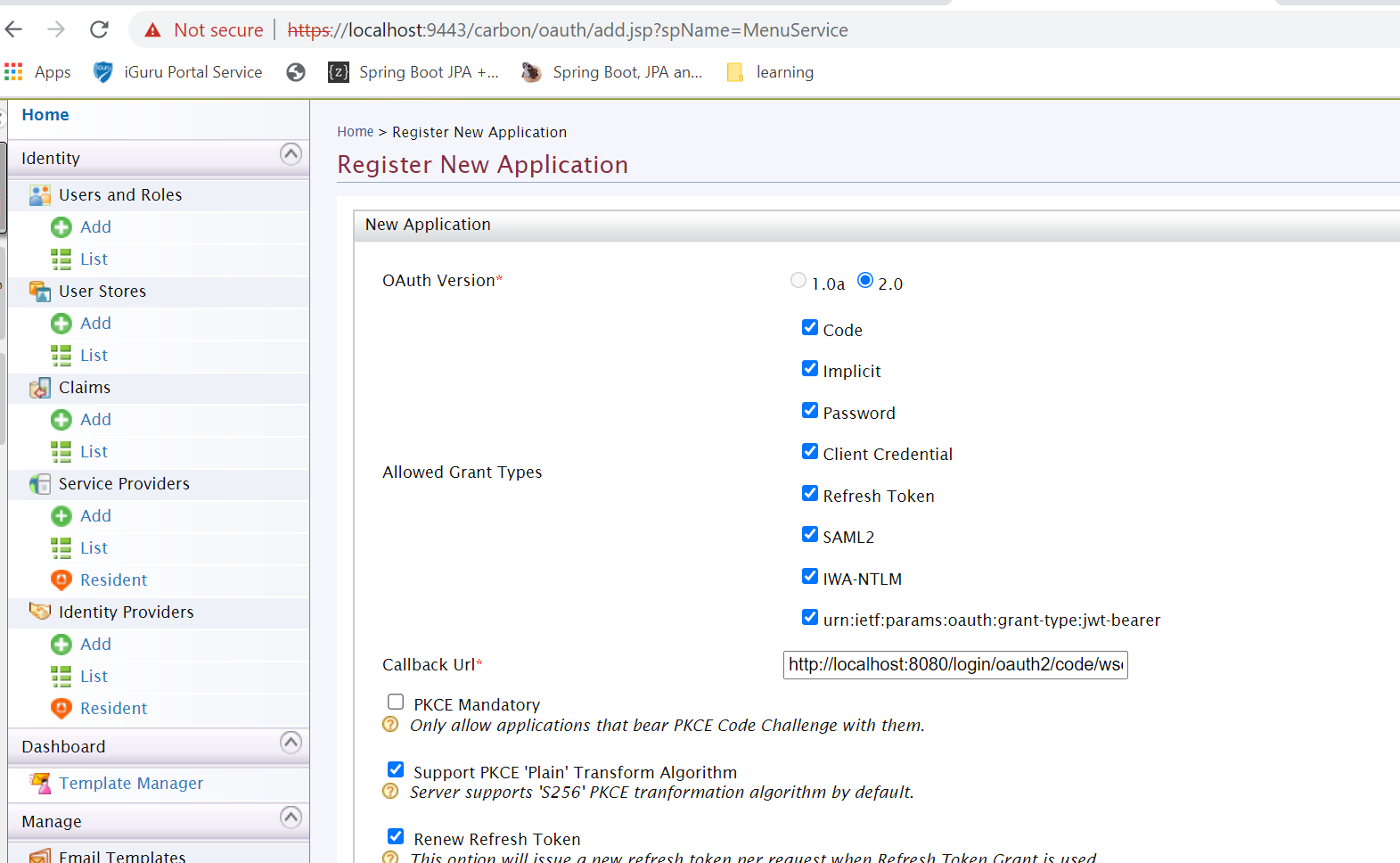


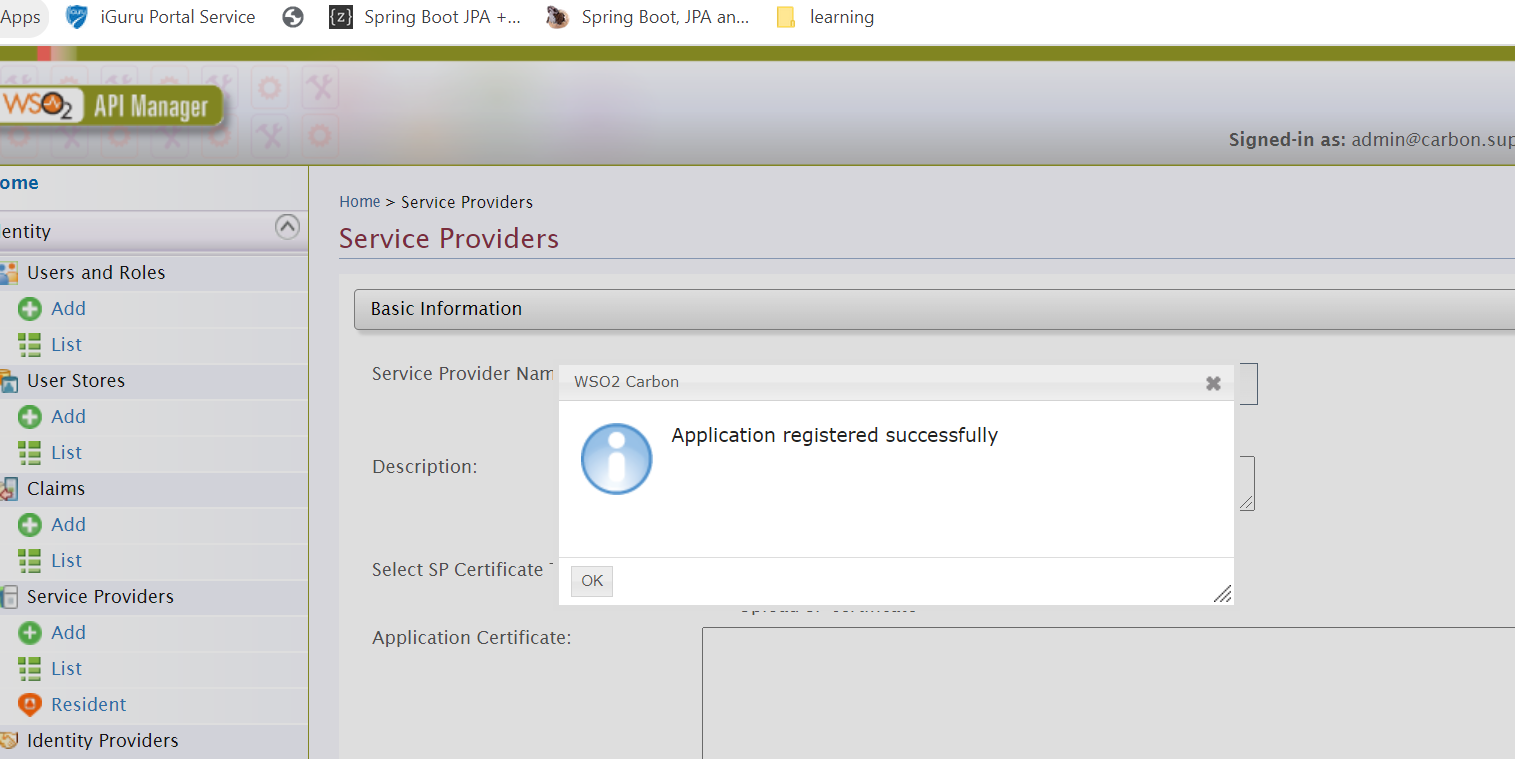
In next window leave all the default values for the fields except call back url. For call back url add the following url as the call back url.

Call back url can be any: regexp=(https://localhost:8080/login/outh2/code/wso2|https://localhost:8080)

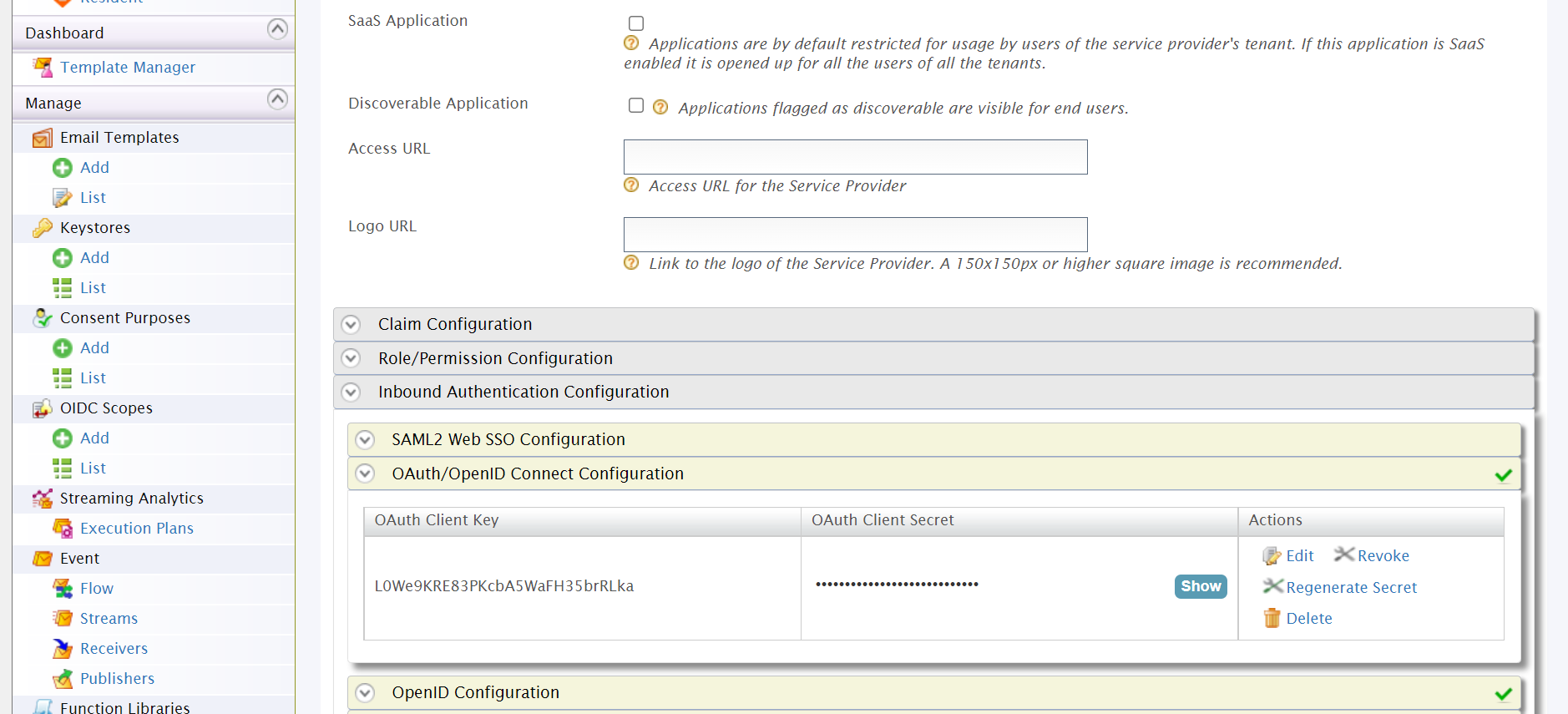
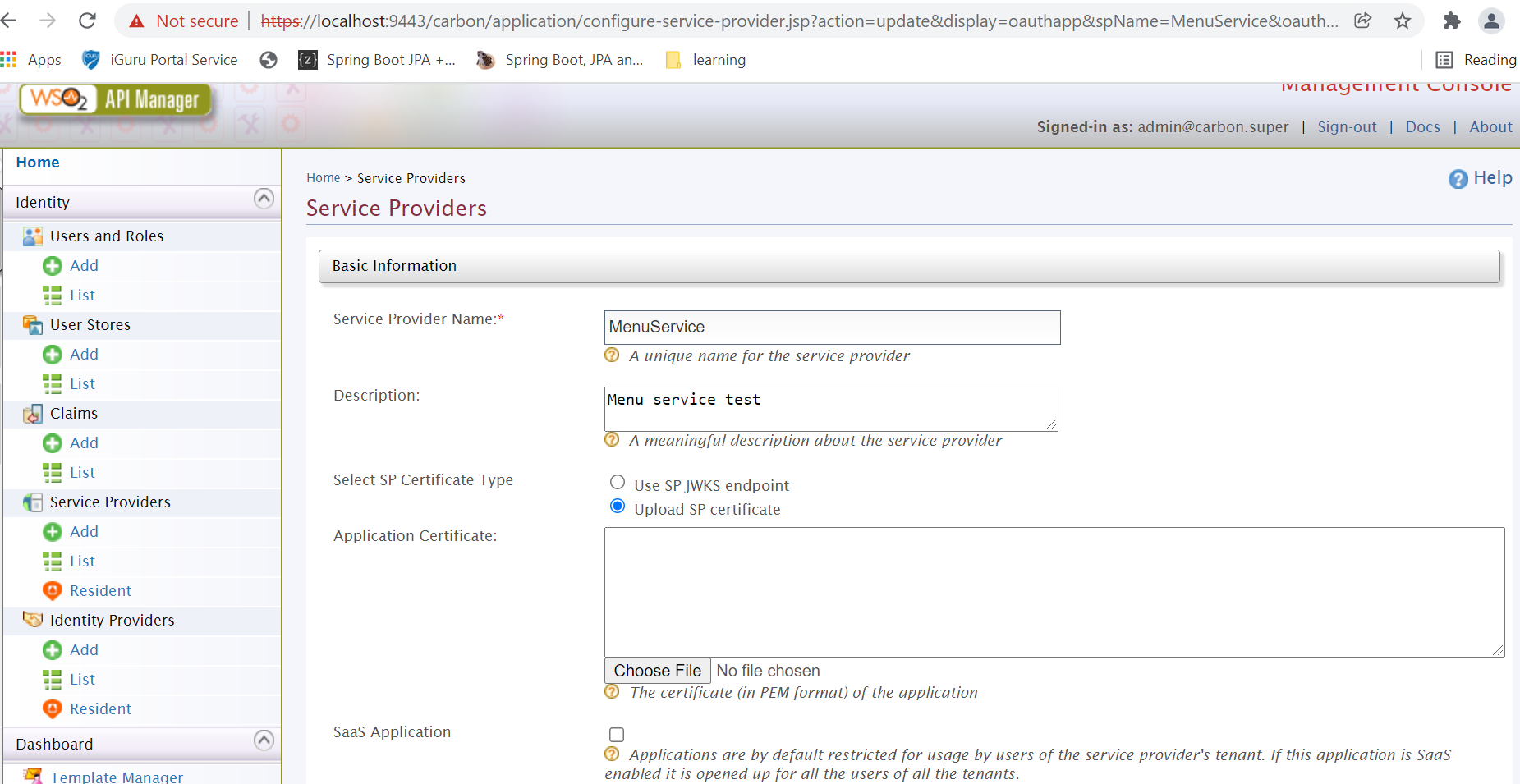
We select below( as not using SSL for now)

<http://localhost:8080/login/oauth2/code/wso2>



It will say, Application Registered successfully, click on ok.

On the service provider page the now the client secret and key has been generated.



You can click on show button to see the actual value of client secret.

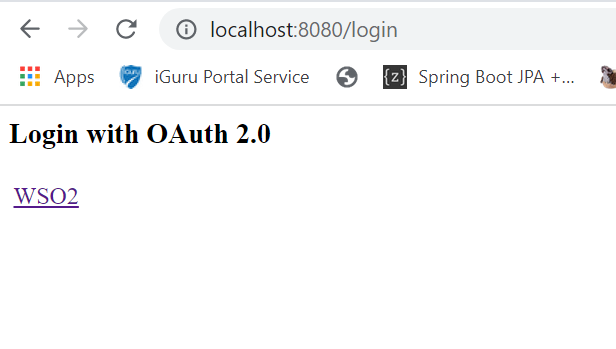
Copy values of client Key and client secret and use it on application.properties file.

Client key : L0We9KRE83PKcbA5WaFH35brRLka

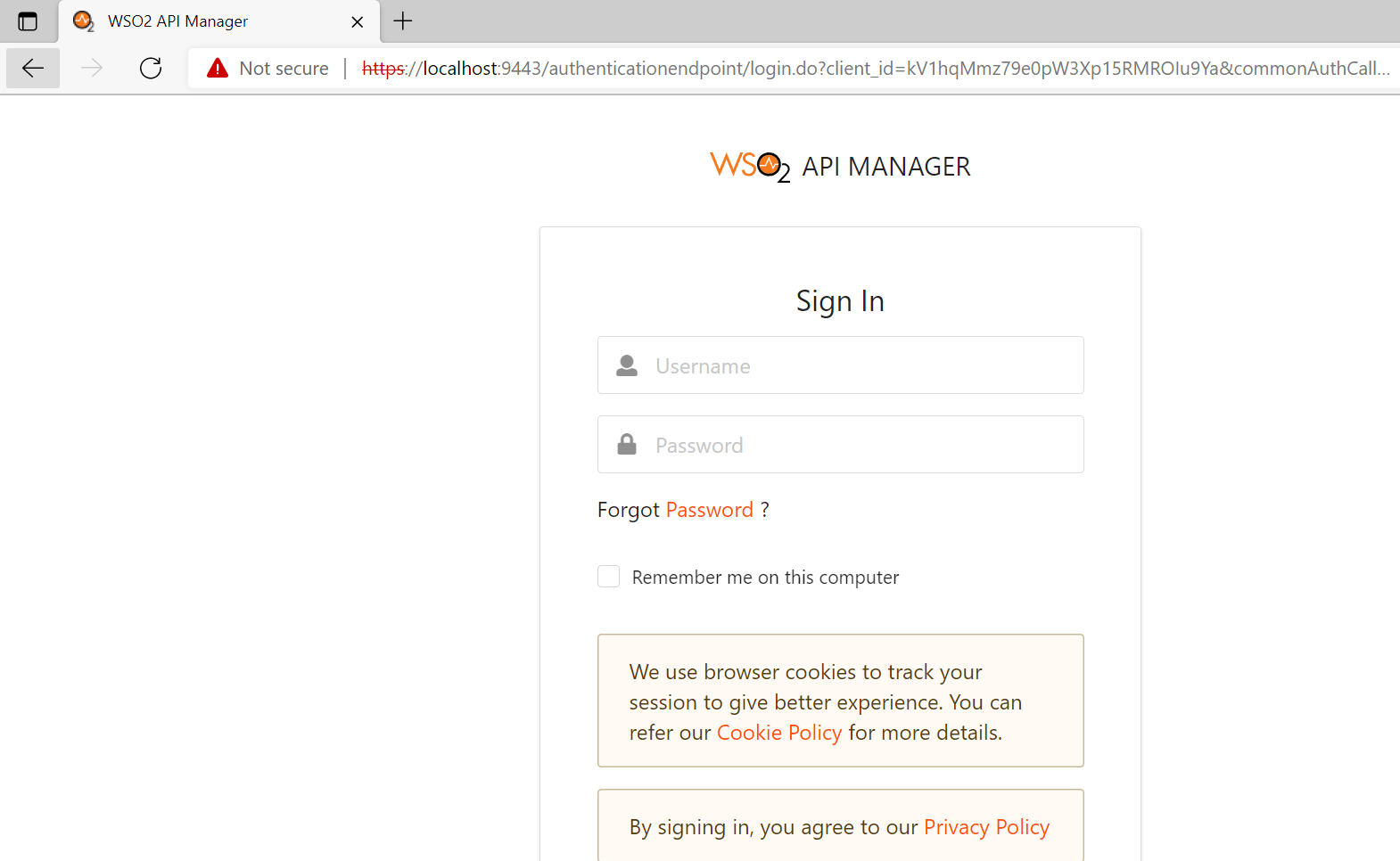
Client secret: MpTtfEWS2x0xZ1zMTKis4W4l2Nia

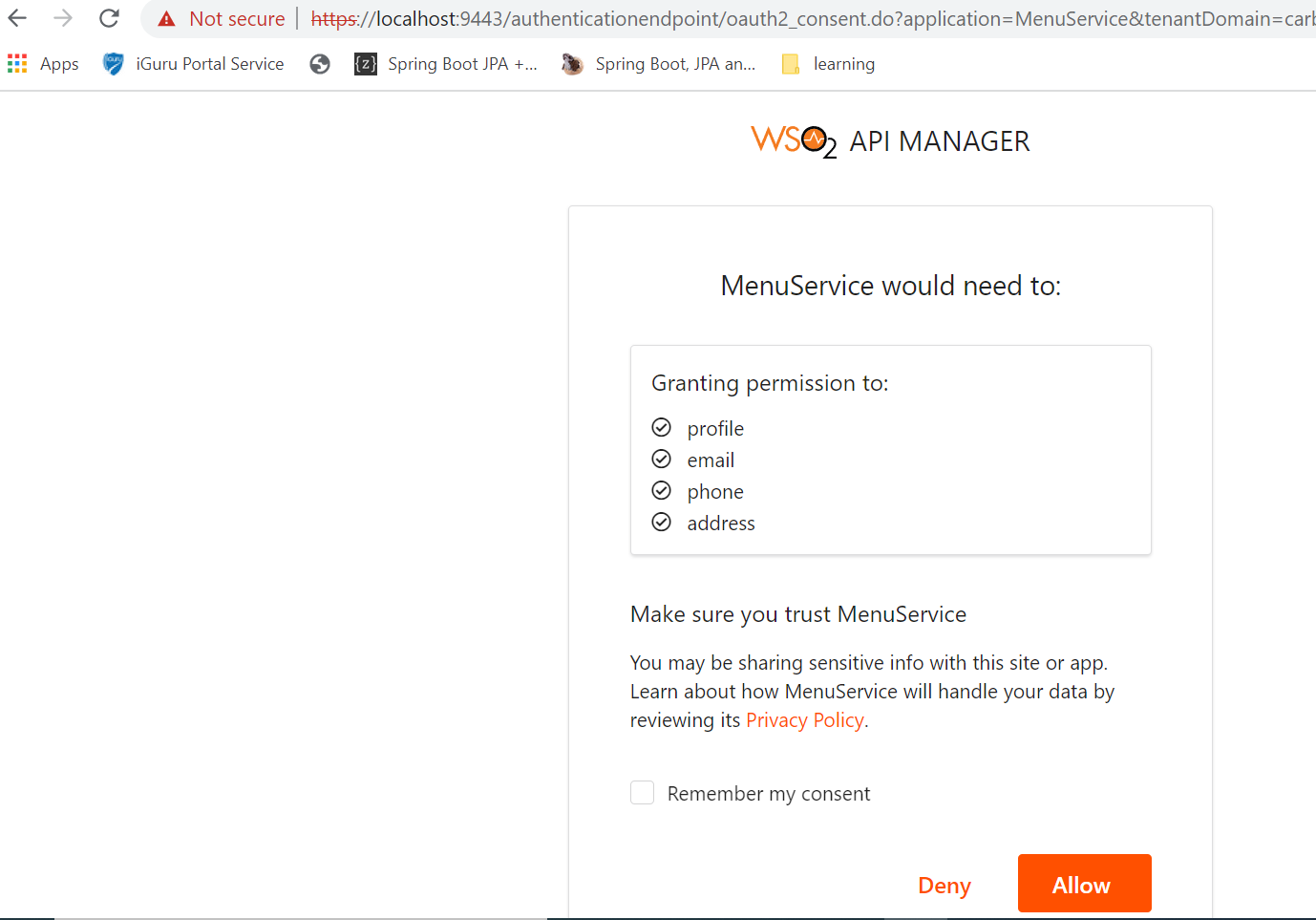
Now start your springboot application and test it by just type <http://localhost:8080> on chrome or IE.

Once your application is up and running you can open it on your localhost and you can see page as below:



While you click on WSO2 link then it will redirect you to wso2 login page where you have to login using registered use, for testing we are login using admin\admin. Once you logged in you can see below page.





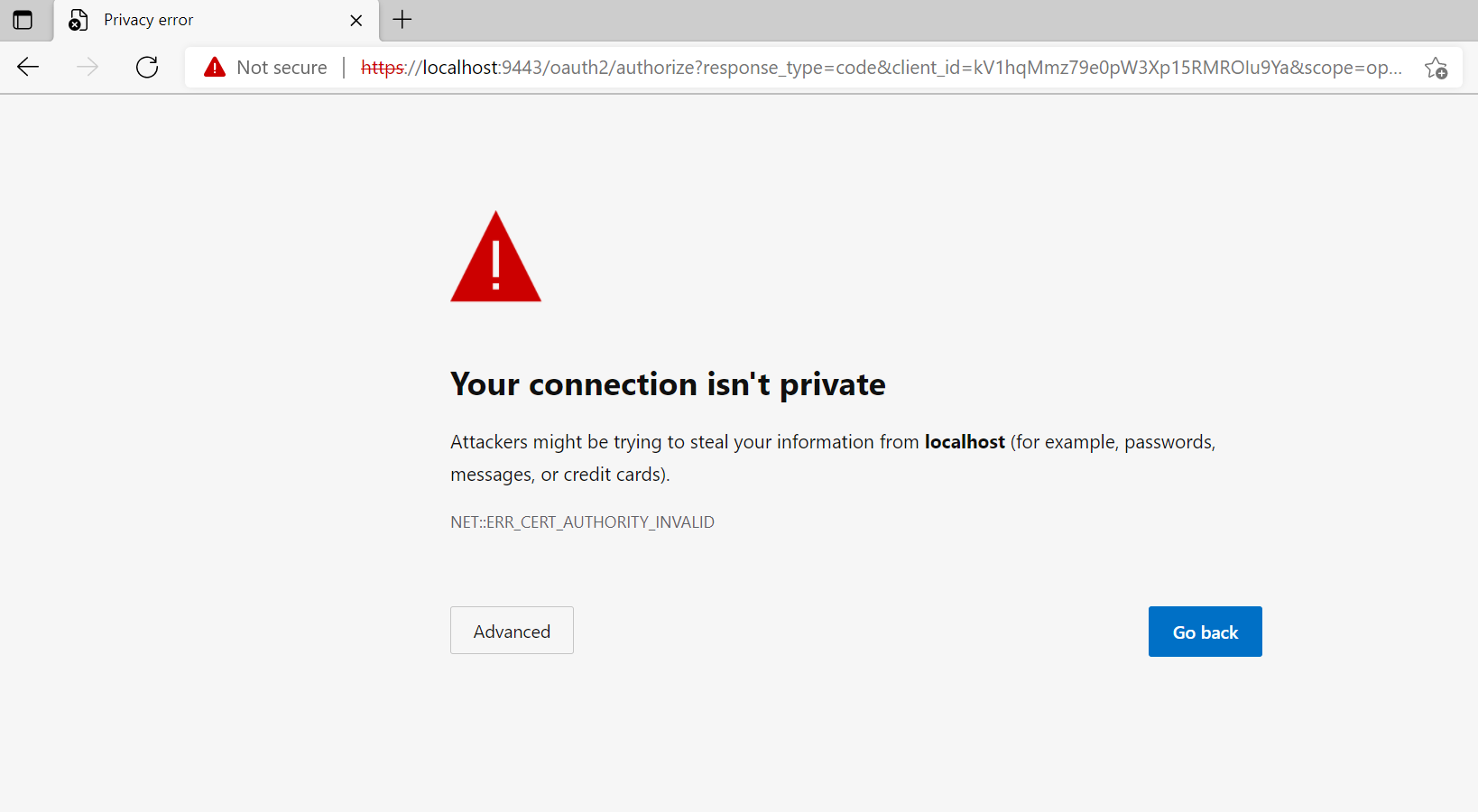
Once you click on Allow you will get redirected and below page will be shown to you :



**Notes**:

1. The spring-boot supports default call-back url to be as *{base-url}/login/oauth2/code/{registration-id}*
2. *The SSL port is 9443 and Http port is 9763.*

If you open the <http://localhost:8080> on IE then you might get the certificate issue, use chrome.



Reference:

<https://medium.com/@balaajanthan/oauth2-login-for-spring-boot-application-with-wso2-identity-server-da0a88893987>

<https://piraveenaparalogarajah.medium.com/secure-your-spring-boot-application-with-wso2-identity-server-8140af8aa30b>

Springboot code is below:

Pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>2.0.0.RELEASE</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.example</groupId>  
 <artifactId>MenuService</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>MenuService</name>  
 <description>Demo project for Spring Boot</description>  
 <properties>  
 <java.version>1.8</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-thymeleaf</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
  
 <!--Additional Dependencies-->  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-config</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-oauth2-client</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-oauth2-jose</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.thymeleaf.extras</groupId>  
 <artifactId>thymeleaf-extras-springsecurity4</artifactId>  
 </dependency>  
  
  
 </dependencies>  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

application.properties

#OAuth Application Properties  
#spring.security.oauth2.client.registration.wso2.client-name=WSO2 Identity Server  
spring.security.oauth2.client.registration.wso2.client-id=hpp1yKogfx7qUbcfS9A5yFFfsWca  
spring.security.oauth2.client.registration.wso2.client-secret=KC4Wwy6RHhgth9LG4LjCc4QlaW8a  
#spring.security.oauth2.client.registration.wso2.redirect-uri={baseUrl}/login/oauth2/code/wso2  
spring.security.oauth2.client.registration.wso2.authorization-grant-type=authorization\_code  
spring.security.oauth2.client.registration.wso2.scope=openid  
  
#Identity Server Properties  
spring.security.oauth2.client.provider.wso2.authorization-uri=http://localhost:9763/oauth2/authorize  
spring.security.oauth2.client.provider.wso2.token-uri=http://localhost:9763/oauth2/token  
spring.security.oauth2.client.provider.wso2.user-info-uri=http://localhost:9763/oauth2/userinfo  
spring.security.oauth2.client.provider.wso2.jwk-set-uri=http://localhost:9763/oauth2/jwks

Controller file

package com.example.MenuService.controller;  
import org.springframework.security.oauth2.client.authentication.OAuth2AuthenticationToken;  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.web.bind.annotation.GetMapping;  
  
@Controller  
public class IndexController {  
 @GetMapping("/")  
 public String getUserName(Model model, OAuth2AuthenticationToken token) {  
  
 model.addAttribute("userName", token.getPrincipal().getName());  
 return "index";  
 }  
}

Security config file is

package com.example.MenuService;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.core.env.Environment;  
import org.springframework.security.oauth2.client.registration.ClientRegistration;  
import org.springframework.security.oauth2.client.registration.ClientRegistrationRepository;  
import org.springframework.security.oauth2.client.registration.InMemoryClientRegistrationRepository;  
import org.springframework.security.oauth2.core.AuthorizationGrantType;  
import org.springframework.security.oauth2.core.ClientAuthenticationMethod;  
import org.springframework.security.oauth2.core.oidc.IdTokenClaimNames;  
  
@Configuration  
public class OAuth2LoginConfig {  
 private static String *CLIENT\_PROPERTY\_KEY* = "spring.security.oauth2.client.registration.wso2.";  
 private static String *PROVIDER\_PROPERTY\_KEY* = "spring.security.oauth2.client.provider.wso2.";  
 @Autowired  
 private Environment env;  
  
 @Bean  
 public ClientRegistrationRepository  
 clientRegistrationRepository() {  
 return new InMemoryClientRegistrationRepository(this.WSO2ClientRegistration());  
 }  
  
 private ClientRegistration WSO2ClientRegistration() {  
  
 return ClientRegistration.*withRegistrationId*("wso2")  
 .clientId(env.getProperty(*CLIENT\_PROPERTY\_KEY* + "client-id"))  
 .clientSecret(env.getProperty(*CLIENT\_PROPERTY\_KEY* + "client-secret"))  
 .clientAuthenticationMethod(ClientAuthenticationMethod.*BASIC*)  
 .authorizationGrantType(AuthorizationGrantType.*AUTHORIZATION\_CODE*)  
 .redirectUriTemplate("{baseUrl}/login/oauth2/code/{registrationId}")  
 .scope("openid", "profile", "email", "address", "phone")  
 .authorizationUri(env.getProperty(*PROVIDER\_PROPERTY\_KEY* + "authorization-uri"))  
 .tokenUri(env.getProperty(*PROVIDER\_PROPERTY\_KEY* + "token-uri"))  
 .userInfoUri(env.getProperty(*PROVIDER\_PROPERTY\_KEY* + "user-info-uri"))  
 .userNameAttributeName(IdTokenClaimNames.*SUB*)  
 .jwkSetUri(env.getProperty(*PROVIDER\_PROPERTY\_KEY* + "jwk-set-uri"))  
 .clientName("WSO2")  
 .build();  
 }  
}

Main application file is

package com.example.MenuService;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class MenuServiceApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(MenuServiceApplication.class, args);  
 }  
  
}

inside resource\templates you need to create **index.html** file and code is below:

<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="http://www.thymeleaf.org">  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>OAuth2 Login </title>  
</head>  
<body>  
<div style="float: right">  
 <div style="float:left">  
 <span style="font-weight:bold">User: <span th:text=${userName}></span> </span>  
 </div>  
 <div style="float:none">&nbsp;</div>  
 <div style="float:right">  
 <form th:action="@{/logout}" method="post">  
 <input type="submit" value="Logout"/>  
 </form>  
 </div>  
</div>  
<h1>OAuth 2.0 Login with Spring Security</h1>  
<div>  
 You are successfully logged in as <span style="font-weight:bold" th:text="${userName}"></span>  
</div>  
</body>  
</html>