**JWT HANDSON**

**Create authentication service that returns JWT**

**Objective:**

* Create authentication controller and configure it in SecurityConfig
* Read Authorization header and decode the username and password
* Generate token based on the user retrieved in the previous step

**PROJECT CODE:**

1. **AuthenticationController.java**

***Folder*:** src/main/java/com/cognizant/SpringJWT/controller/ AuthenticationController.java

**CODE:**

package com.cognizant.SpringJWT.controller;

import com.cognizant.SpringJWT.util.JwtUtil;

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

import org.springframework.http.ResponseEntity;

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

import javax.servlet.http.HttpServletRequest;

import java.util.Base64;

@RestController

public class AuthenticationController {

@Autowired

private JwtUtil;

@GetMapping("/authenticate")

public ResponseEntity<?> authenticate(HttpServletRequest request) {

String authHeader = request.getHeader("Authorization");

if (authHeader != null && authHeader.startsWith("Basic ")) {

String base64Credentials = authHeader.substring("Basic ".length());

byte[] decoded = Base64.getDecoder().decode(base64Credentials);

String credentials = new String(decoded);

String[] parts = credentials.split(":", 2);

String username = parts[0];

String password = parts[1];

// Validate credentials (from in-memory setup)

if (("user".equals(username) || "admin".equals(username)) && "pwd".equals(password)) {

String token = jwtUtil.generateToken(username);

return ResponseEntity.ok("{\"token\":\"" + token + "\"}");

}

}

return ResponseEntity.status(401).body("{\"error\": \"Invalid Credentials\"}");

}

}

1. **CountryController.java**

***Folder:*** src/main/java/com/cognizant/SpringJWT/controller/CountryController.java

**CODE:**

package com.cognizant.SpringJWT.controller;

import com.cognizant.SpringJWT.model.Country;

import com.cognizant.SpringJWT.service.CountryService;

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

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.web.bind.annotation.GetMapping;

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

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

import org.springframework.web.bind.annotation.RestController;

import java.util.ArrayList;

@RestController

@RequestMapping("/")

public class CountryController {

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

Country = (Country) context.getBean("country");

ArrayList<Country> countries = (ArrayList<Country>) context.getBean("countryList");

@GetMapping("/country")

public Country getCountryIndia() {

return country;

}

@GetMapping("/countries")

public ArrayList<Country> getAllCountries(){

return countries;

}

@Autowired

private CountryService;

@GetMapping("/countries/{code}")

public Country getCountry(@PathVariable String code) {

return countryService.getCountry(code);

}

}

1. **Country.java**

***Folder*:** src/main/java/com/cognizant/SpringJWT/model/Country.java

**CODE:**

package com.cognizant.SpringJWT.model;

public class Country {

private String code;

private String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

@Override

public String toString() {

return "Country{" +

"code='" + code + '\'' +

", name='" + name + '\'' +

'}';

}

}

1. **SecurityConfig.java**

***Folder*:** src/main/java/com/cognizant/SpringJWT/security/SecurityConfig.java

**CODE:**

package com.cognizant.SpringJWT.security;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.annotation.Bean;

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;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

private static final Logger LOGGER = LoggerFactory.getLogger(SecurityConfig.class);

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.inMemoryAuthentication()

.withUser("admin").password(passwordEncoder().encode("pwd")).roles("ADMIN")

.and()

.withUser("user").password(passwordEncoder().encode("pwd")).roles("USER");

}

@Bean

public PasswordEncoder passwordEncoder() {

LOGGER.info("Initializing PasswordEncoder bean");

return new BCryptPasswordEncoder();

}

// @Override

// protected void configure(HttpSecurity httpSecurity) throws Exception {

// httpSecurity.csrf().disable().httpBasic().and()

// .authorizeRequests().antMatchers("/countries").hasRole("USER");

// }

@Override

protected void configure(HttpSecurity httpSecurity) throws Exception {

httpSecurity

.csrf().disable()

.httpBasic()

.and()

.authorizeRequests()

.antMatchers("/authenticate").permitAll()

.antMatchers("/countries").hasRole("USER")

.anyRequest().authenticated();

}

}

1. **CountryService.java**

***Folder*:** src/main/java/com/cognizant/SpringJWT/service/CountryService.java

**CODE:**

package com.cognizant.SpringJWT.service;

import com.cognizant.SpringJWT.model.Country;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service

public class CountryService {

public Country getCountry(String code) {

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

ArrayList<Country> countryList = (ArrayList<Country>) context.getBean("countryList");

return countryList.stream()

.filter(c -> c.getCode().equalsIgnoreCase(code))

.findFirst()

.orElseThrow(() -> new RuntimeException("Country not found"));

}

}

1. **JwtUtil.java**

***Folder:*** src/main/java/com/cognizant/SpringJWT/util/JwtUtil.java

**CODE:**

package com.cognizant.SpringJWT.util;l

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import javax.crypto.spec.SecretKeySpec;

import java.security.Key;

import java.util.Date;

import java.util.Base64;

@Component

public class JwtUtil {

private final String SECRET\_KEY = "secretkey1234567890secretkey1234567890"; // at least 256-bit

private final Key key = new SecretKeySpec(Base64.getDecoder().decode(Base64.getEncoder().encodeToString(SECRET\_KEY.getBytes())),

SignatureAlgorithm.HS256.getJcaName());

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date(System.currentTimeMillis()))

.setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 10)) // 10 minutes

.signWith(key)

.compact();

}

}

1. **SpringJwtApplication.java**

***Folder*:** src/main/java/com/cognizant/SpringJWT/SpringJwtApplication.java

**CODE:**

package com.cognizant.SpringJWT;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringJwtApplication {

public static void main(String[] args) {

SpringApplication.run(SpringJwtApplication.class, args);

}

}

8) application.properties

Folder: src/main/resources/application.properties

Code:

spring.application.name=SpringJWT

server.port=8090

1. **country.xml**

***Folder*:** src/main/resources/country.xml

**CODE:**

<?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.xsd">

<!-- bean definitions here -->

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean id="country" class="com.cognizant.SpringJWT.model.Country">

<property name="code" value="IN"/>

<property name="name" value="India"/>

</bean>

<bean class="com.cognizant.SpringJWT.model.Country">

<property name="code" value="US"/>

<property name="name" value="United States"/>

</bean>

<bean class="com.cognizant.SpringJWT.model.Country">

<property name="code" value="JP"/>

<property name="name" value="Japan"/>

</bean>

<bean class="com.cognizant.SpringJWT.model.Country">

<property name="code" value="DE"/>

<property name="name" value="Germany"/>

</bean>

</list>

</constructor-arg>

</bean>

<bean id="country" class="com.cognizant.SpringJWT.model.Country">

<property name="code" value="IN"/>

<property name="name" value="India"/>

</bean>

</beans>

1. **SprintJwtApplicationTests.java**

***Folder*:** src/test/java/com/cognizant/SpringJWT/SprintJwtApplicationTests.java

**CODE:**

package com.cognizant.SpringJWT;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class SprintJwtApplicationTests {

@Test

void contextLoads() {

}

}

1. **pom.xml**

**CODE:**

<?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>2.6.6</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.cognizant</groupId>

<artifactId>SpringJWT</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>SpringJWT</name>

<description>Demo project for Spring Boot</description>

<url/>

<licenses>

<license/>

</licenses>

<developers>

<developer/>

</developers>

<scm>

<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<properties>

<java.version>17</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-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</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>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.11.5</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId>

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**OUTPUT:**



