**Rubina S SupersetID:6383615**

**Week 4**

**Spring REST using SpringBoot**

**Hands-on 1**

**Create a Spring Web Project using Maven**

Steps Followed:

1. Go to<https://start.spring.io/>
2. Change Group as “com.cognizant”
3. Change Artifact Id as “spring-learn”
4. Select Spring Boot DevTools and Spring Web
5. Create and download the project as zip
6. Extract the zip in root folder to Eclipse Workspace
7. Build the project using ‘mvn clean package

-Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050

-Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050

-Dhttp.proxyUser=123456’ command in command line

1. Import the project in Eclipse "File > Import > Maven > Existing Maven Projects > Click Browse and select extracted folder > Finish"
2. Include logs to verify if main() method of SpringLearnApplication. Run the SpringLearnApplication class.

**CODE-SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) { LOGGER.info("START");

SpringApplication.run(SpringLearnApplication.class, args); LOGGER.info("END");

}

}

# OUTPUT

****

**Hands-on 2**

**Spring Core – Load Country from Spring Configuration XML**

Steps followed:

Created a Spring Boot project with Spring Web and DevTools. Created a Country class with a name property.

Added country.xml in src/main/resources and defined a Country bean.

Loaded the XML using ClassPathXmlApplicationContext in displayCountry() method.

Called displayCountry() from main() to print the loaded country.

1. **Country.java**

package com.cognizant.spring\_learn;

public class Country { private String name;

public Country() {} public String getName() {

return name;

}

public void setName(String name) { this.name = name;

}

@Override

public String toString() {

return "Country [name=" + name + "]";

}

}

1. **Country.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>

https://[www.springframework.org/schema/beans/spring-beans.xsd](http://www.springframework.org/schema/beans/spring-beans.xsd)">

<bean id="country" class="com.cognizant.spring\_learn.Country">

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

</bean>

</beans>

1. **SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication; import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) { LOGGER.info("START");

SpringApplication.run(SpringLearnApplication.class, args); displayCountry();

LOGGER.info("END");

}

public static void displayCountry() { ApplicationContext context = new

ClassPathXmlApplicationContext("country.xml");

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

System.out.println("Country Loaded: " + country); ((ClassPathXmlApplicationContext) context).close();

}

}

# OUTPUT

****

**Hands-on 3**

**Hello World RESTful Web Service**

Write a REST service in the spring learn application created earlier, that returns the text "Hello World!!" using Spring Web Framework. Refer details below:

**Method:** GET

**URL:** /hello

**Controller:** com.cognizant.spring-learn.controller.HelloController

**Method Signature:** public String sayHello()

**Method Implementation:** return hard coded string "Hello World!!"

**Sample Request**: <http://localhost:8083/hello>

**Sample Response:** Hello World!!

# SOLUTION

**Add the REST Controller (HelloController)** **HelloController.java**

package com.cognizant.spring\_learn.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

@RestController

public class HelloController {

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

@GetMapping("/hello") public String sayHello() {

LOGGER.info("START - sayHello()"); String response = "Hello World!!"; LOGGER.info("END - sayHello()"); return response;

}

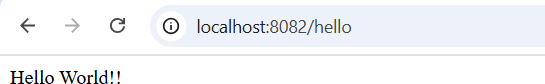
}

# OUTPUT

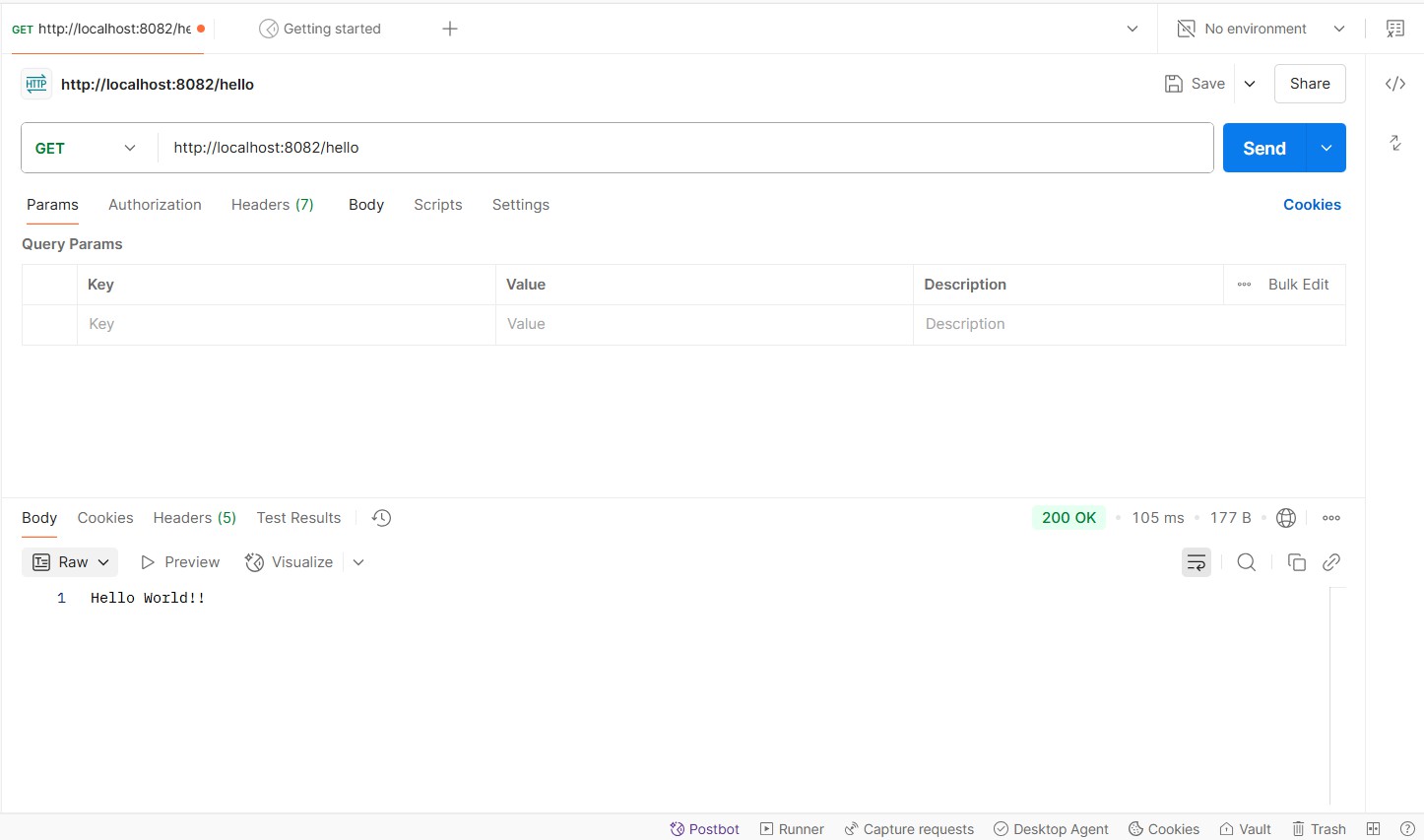
**Console Output in Eclipse IDE**

****

**API Output in Web Browser**

****

**API Response in Postman**

****

**Hands-on 4**

**REST - Country Web Service**

Write a REST service that returns India country details in the earlier created spring learn application.

**URL**: /country

**Controller**: com.cognizant.spring-learn.controller.CountryController

**Method Annotation**: @RequestMapping

**Method Name**: getCountryIndia()

**Method Implementation**: Load India bean from spring xml configuration and return

**Sample Request**: <http://localhost:8083/country>

**Sample Response**:

{

"code": "IN",

"name": "India"

}

# SOLUTION

1. **Country.java – Model Class**

package com.cognizant.spring\_learn;

public class Country { private String code; private String name;

public String getCode() { return code;

}

public void setCode(String code) { this.code = code;

}

public String getName() { return name;

}

public void setName(String name) { this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

1. **Country.xml – Spring Configuration File**

<?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 id="in" class="com.cognizant.spring\_learn.Country">

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

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

</bean>

</beans>

1. **CountryController.java – Controller Class**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.Country; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext; import org.springframework.web.bind.annotation.RequestMapping;

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

@RestController

public class CountryController {

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

@RequestMapping("/country") public Country getCountryIndia() {

LOGGER.info("START - getCountryIndia()"); ApplicationContext context = new

ClassPathXmlApplicationContext("country.xml"); Country country = (Country) context.getBean("in"); LOGGER.info("END - getCountryIndia()");

return country;

}

}

1. **SpringLearnApplication.java – Main Class**

package com.cognizant.spring\_learn; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) { LOGGER.info("START");

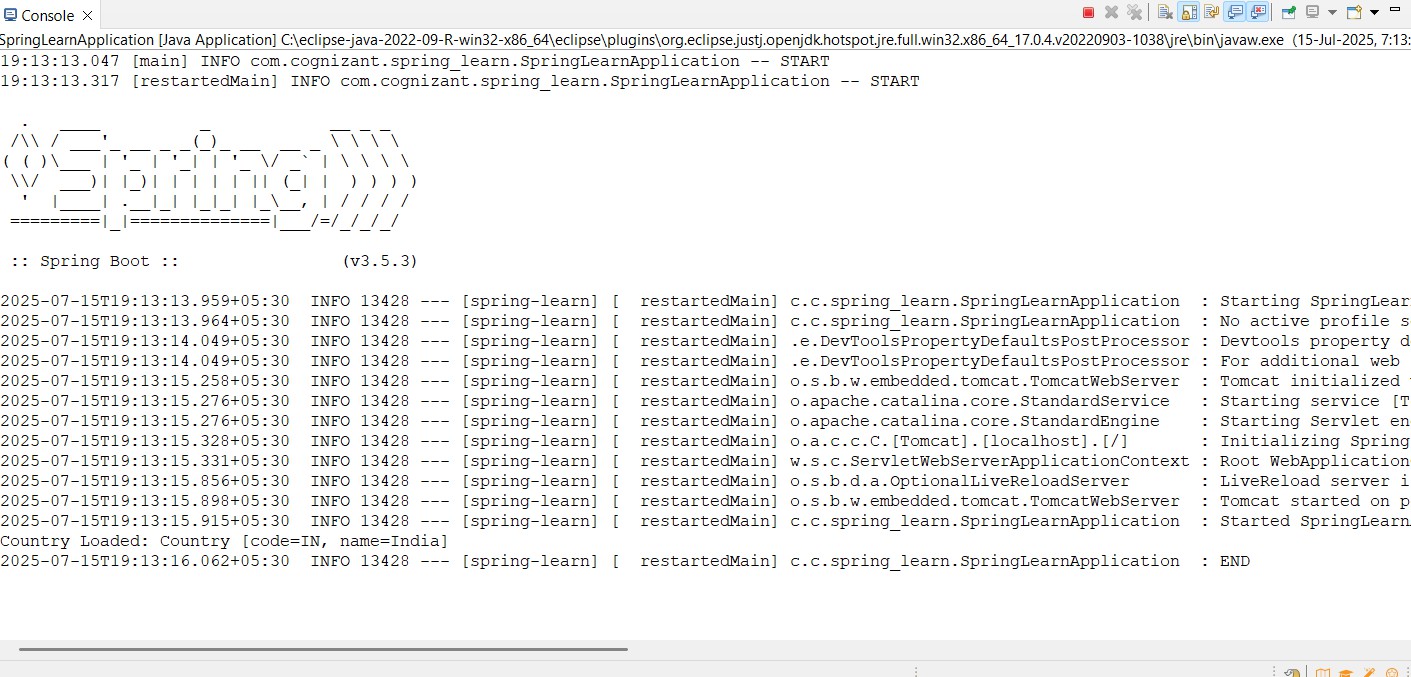
SpringApplication.run(SpringLearnApplication.class, args); LOGGER.info("END");

}

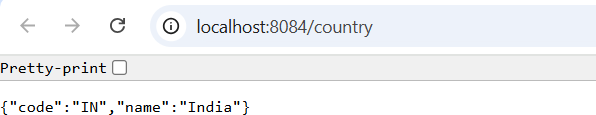
}

# OUTPUT

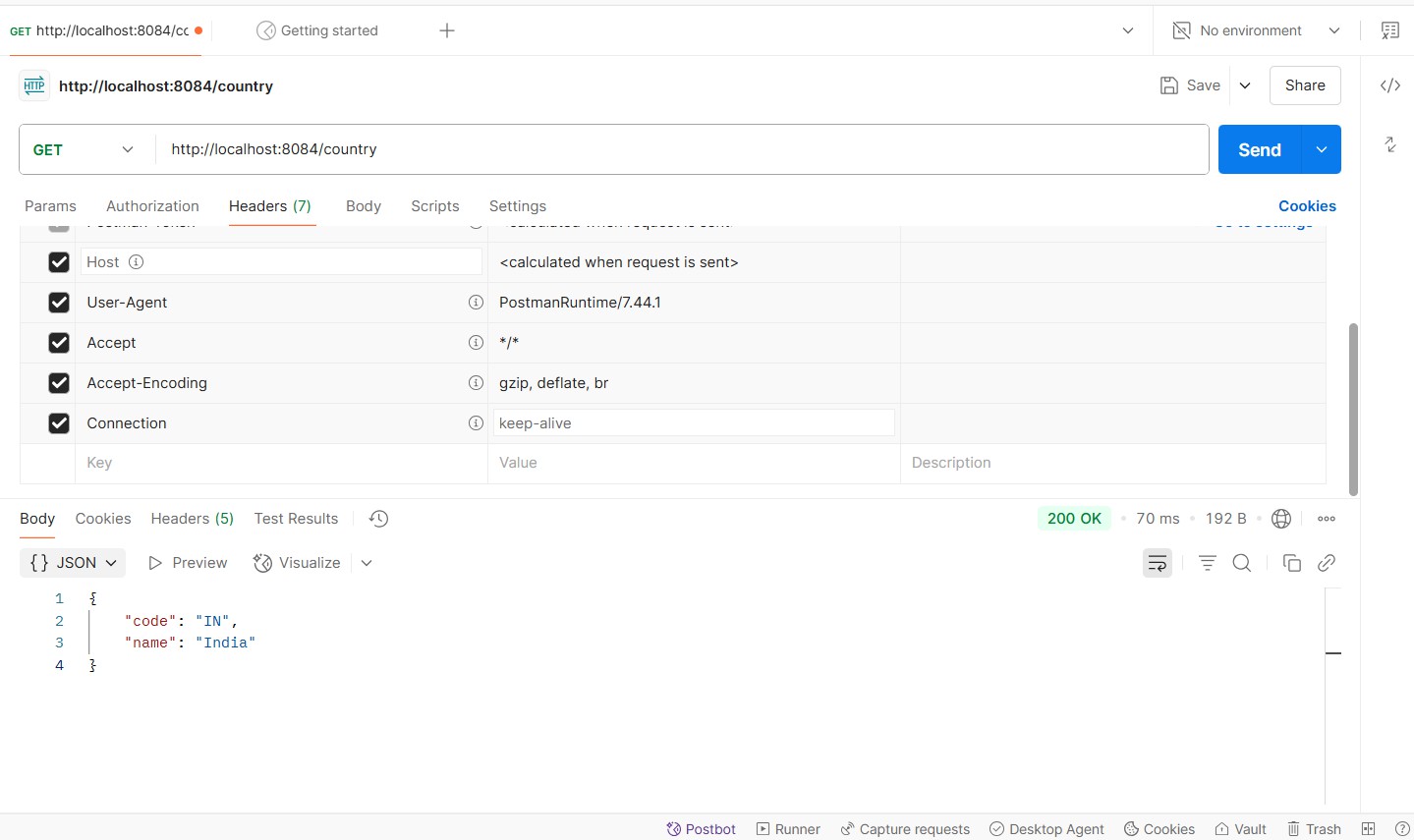
**Console Output in Eclipse IDE**

****

**API Output in Web Browser**

****

**API Response in Postman**

****

**JWT Handson**

**Create authentication service that returns JWT**

As part of first step of JWT process, the user credentials needs to be sent to authentication service request that generates and returns the JWT.

Ideally when the below curl command is executed that calls the new authentication service, the token should be responded. Kindly note that the

credentials are passed using -u option.

**Request**

curl -s -u user:pwd <http://localhost:8090/authenticate>

**Response**

{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5ND

c0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-hwKfoqZYlaVQqEUiBloWcWn0f t3tgv0dL0"}

This can be incorporated as three major steps:

* 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

# SOLUTION

1. **Create authentication controller and configure it in SecurityConfig AuthenticationController.java**

package com.cognizant.spring\_learn.controller; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import java.util.HashMap; import java.util.Map;

@RestController

public class AuthenticationController { private static final Logger LOGGER =

LoggerFactory.getLogger(AuthenticationController.class); @GetMapping("/authenticate")

public Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {

LOGGER.info("START - /authenticate"); LOGGER.debug("Authorization Header: {}", authHeader);

Map<String, String> map = new HashMap<>(); map.put("token", "");

LOGGER.info("END - /authenticate"); return map;

}

}

**SecurityConfig.java**

package com.cognizant.spring\_learn.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity; import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.provisioning.InMemoryUserDetailsManager; import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class SecurityConfig { @Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception { http

.csrf().disable()

.authorizeHttpRequests()

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

.requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

.anyRequest().authenticated()

.and()

.httpBasic();

return http.build();

}

@Bean

public InMemoryUserDetailsManager userDetailsService() { UserDetails user = User.withDefaultPasswordEncoder()

.username("user")

.password("pwd")

.roles("USER")

.build();

return new InMemoryUserDetailsManager(user);

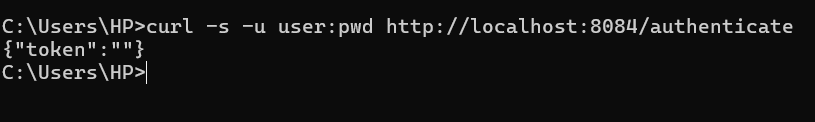
}

}

**Console Output in Eclipse IDE**

****

**API Response in Terminal**

****

1. **Read Authorization header and decode the username and password AuthenticationController.java**

package com.cognizant.spring\_learn.controller; import java.nio.charset.StandardCharsets; import java.util.Date;

import java.util.HashMap; import java.util.Map;

import javax.crypto.SecretKey; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Value; import org.springframework.security.core.Authentication;

import org.springframework.security.core.context.SecurityContextHolder; import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController; import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.JwtBuilder;

import io.jsonwebtoken.SignatureAlgorithm; import io.jsonwebtoken.security.Keys; @RestController

public class AuthenticationController {

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

@Value("${jwt.secret}") private String secretKey; @GetMapping("/authenticate")

public Map<String, String> authenticate() {

LOGGER.info("START - /authenticate"); Authentication authentication =

SecurityContextHolder.getContext().getAuthentication(); String user = authentication.getName(); LOGGER.info("Authenticated user: {}", user); String token = generateJwt(user); LOGGER.info("Generated Token: {}", token); Map<String, String> response = new HashMap<>(); response.put("token", token);

LOGGER.info("END - /authenticate"); return response;

}

private String generateJwt(String user) {

SecretKey key = Keys.hmacShaKeyFor(secretKey.getBytes(StandardCharsets.UTF\_8));

JwtBuilder builder = Jwts.builder()

.setSubject(user)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 20 \* 60 \* 1000))

.signWith(key, SignatureAlgorithm.HS256);

return builder.compact();

}

}

**SecurityConfig.java**

package com.cognizant.spring\_learn.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity; import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.provisioning.InMemoryUserDetailsManager; import org.springframework.security.web.SecurityFilterChain;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception { http

.csrf().disable()

.authorizeHttpRequests()

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

.requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

.anyRequest().authenticated()

.and()

.httpBasic();

return http.build();

}

@Bean

public InMemoryUserDetailsManager userDetailsService() { UserDetails user = User.withDefaultPasswordEncoder()

.username("user")

.password("pwd")

.roles("USER")

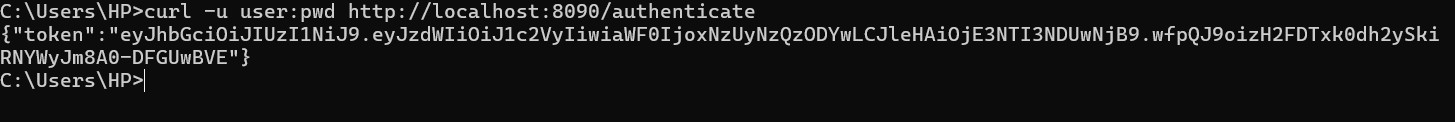
.build();

return new InMemoryUserDetailsManager(user);

}

}

# OUTPUT

****

1. **Generate token based on the user SOLUTION**

**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>3.5.3</version>

<relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learn</name>

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

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<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.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</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>

**AuthenticationController.java**

package com.cognizant.spring\_learn.controller;

import com.cognizant.spring\_learn.model.JwtRequest; import com.cognizant.spring\_learn.model.JwtResponse; import com.cognizant.spring\_learn.util.JwtUtil;

import org.springframework.http.ResponseEntity; import org.springframework.web.bind.annotation.\*; @RestController

public class AuthenticationController { @GetMapping("/test")

public String test() {

return "API is working!";

}

@PostMapping("/authenticate")

public ResponseEntity<?> createAuthenticationToken(@RequestBody JwtRequest jwtRequest) {

if ("admin".equals(jwtRequest.getUsername()) && "admin123".equals(jwtRequest.getPassword())) {

String token = JwtUtil.generateToken(jwtRequest.getUsername()); return ResponseEntity.ok(new JwtResponse(token));

} else {

return ResponseEntity.status(401).body("Invalid credentials");

}

}

}

**JwtRequest.java**

package com.cognizant.spring\_learn.model;

public class JwtRequest { private String username; private String password;

public JwtRequest() {}

public JwtRequest(String username, String password) { this.username = username;

this.password = password;

}

public String getUsername() { return username;

}

public void setUsername(String username) { this.username = username;

}

public String getPassword() { return password;

}

public void setPassword(String password) { this.password = password;

}

}

**JwtResponse.java**

package com.cognizant.spring\_learn.model;

public class JwtResponse { private String token;

public JwtResponse(String token) { this.token = token;

}

public String getToken() { return token;

}

public void setToken(String token) { this.token = token;

}

}

**JwtUtil.java**

package com.cognizant.spring\_learn.util;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm; import java.util.Date;

public class JwtUtil {

private static final String SECRET\_KEY = "secretkey"; public static String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 1200000))

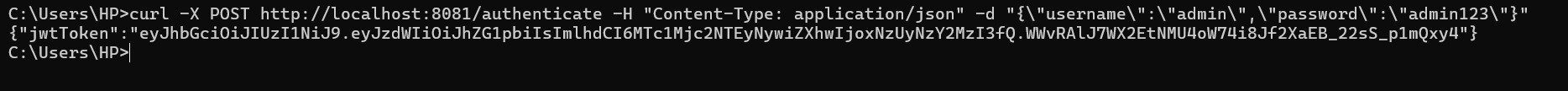
.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

**OUTPUT**

****