WEEK 4

Mandatory Hands-On

**JWT Hands-On**

**Create authentication service that returns JWT**

**Creating Security Config class:**

package com.cognizant.jwt.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.web.SecurityFilterChain;  
  
@Configuration  
public class SecurityConfig {  
  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
 http.csrf().disable()  
 .authorizeHttpRequests()  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated();  
 return http.build();  
 }  
}

package com.cognizant.jwt.controller;  
  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import org.springframework.web.bind.annotation.\*;  
  
import javax.servlet.http.HttpServletRequest;  
import java.util.Base64;  
import java.util.Date;  
  
@RestController  
public class AuthenticationController {  
  
 private static final String *SECRET\_KEY* = "mySecretKey";  
  
 @RequestMapping("/authenticate")  
 public String authenticate(HttpServletRequest request) {  
 String header = request.getHeader("Authorization");  
  
 if (header == null || !header.startsWith("Basic ")) {  
 return "Missing Authorization header";  
 }  
  
 String base64Credentials = header.substring("Basic ".length());  
 String credentials = new String(Base64.*getDecoder*().decode(base64Credentials));  
 String[] values = credentials.split(":", 2);  
  
 String username = values[0];  
 String password = values[1];  
  
 if ("user".equals(username) && "pwd".equals(password)) {  
 String token = Jwts.*builder*()  
 .setSubject(username)  
 .setIssuedAt(new Date())  
 .setExpiration(new Date(System.*currentTimeMillis*() + 1000 \* 60 \* 10)) // 10 min  
 .signWith(SignatureAlgorithm.*HS256*, *SECRET\_KEY*)  
 .compact();  
 return "{\"token\":\"" + token + "\"}";  
 } else {  
 return "Invalid credentials";  
 }  
 }  
}

**Creating Spring JWT Authentication class:**

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

server.port=8090  
logging.level.org.springframework=INFO

**Updated pom.xml file:**

<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/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.cognizant</groupId>  
 <artifactId>jwt</artifactId>  
 <version>1.0.0</version>  
 <packaging>jar</packaging>  
  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.2.0</version>  
 </parent>  
  
 <dependencies>  
 <!-- Web & Security -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
 </dependency>  
  
 <!-- JWT -->  
 <dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt</artifactId>  
 <version>0.9.1</version>  
 </dependency>  
  
 <!-- Lombok (optional) -->  
 <dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.30</version>  
 <scope>provided</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
</project>

**Sample Output:**

