**CREATE AUTHENTICATION SERVICE THAT RETURNS JWT**

**APPROACH**

**· Create a Spring Boot Maven project with Web, Security, and JJWT dependencies.**

**· Configure SecurityConfig with SecurityFilterChain and in-memory user details.**

**· Implement AuthenticationController to generate JWT using Keys.secretKeyFor(SignatureAlgorithm.HS256).**

**· Set application.properties (e.g., spring.application.name, server.port=8083).**

**· Run the project and test the /authenticate endpoint using Basic Authentication to receive a JWT token.**

**CODE**

**AuthenticationController.java  
 package com.cognizant.spring\_learn.controller;**

**import io.jsonwebtoken.Jwts;**

**import io.jsonwebtoken.SignatureAlgorithm;**

**import io.jsonwebtoken.security.Keys;**

**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.security.Key;**

**import java.util.Base64;**

**import java.util.Date;**

**import java.util.HashMap;**

**import java.util.Map;**

**@RestController**

**public class AuthenticationController {**

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

**private final Key key = Keys.secretKeyFor(SignatureAlgorithm.HS256);**

**@GetMapping("/authenticate")**

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

**LOGGER.info("START authenticate()");**

**String user = getUser(authHeader);**

**String token = generateJwt(user);**

**Map<String, String> map = new HashMap<>();**

**map.put("token", token);**

**LOGGER.info("END authenticate()");**

**return map;**

**}**

**private String getUser(String authHeader) {**

**LOGGER.info("START getUser()");**

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

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

**String credentials = new String(credDecoded);**

**String user = credentials.split(":")[0];**

**LOGGER.info("Decoded User: {}", user);**

**LOGGER.info("END getUser()");**

**return user;**

**}**

**private String generateJwt(String user) {**

**LOGGER.info("START generateJwt()");**

**String token = Jwts.builder()**

**.setSubject(user)**

**.setIssuedAt(new Date())**

**.setExpiration(new Date(System.currentTimeMillis() + 1200000)) // 20 minutes**

**.signWith(key)**

**.compact();**

**LOGGER.info("END generateJwt()");**

**return token;**

**}**

**}**

**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.config.annotation.web.configuration.EnableWebSecurity;**

**import org.springframework.security.core.userdetails.User;**

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

**import org.springframework.security.core.userdetails.UserDetailsService;**

**import org.springframework.security.provisioning.InMemoryUserDetailsManager;**

**import org.springframework.security.web.SecurityFilterChain;**

**@Configuration**

**@EnableWebSecurity**

**public class SecurityConfig {**

**@Bean**

**public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {**

**http.csrf().disable()**

**.authorizeHttpRequests(authz -> authz**

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

**.anyRequest().authenticated())**

**.httpBasic();**

**return http.build();**

**}**

**@Bean**

**public UserDetailsService userDetailsService() {**

**UserDetails user = User.withDefaultPasswordEncoder()**

**.username("user")**

**.password("pwd")**

**.roles("USER")**

**.build();**

**return new InMemoryUserDetailsManager(user);**

**}**

**}**

**application.properties**

**server.port=8083**

**SpringLearnApplication.java  
 package com.cognizant.spring\_learn;**

**import org.springframework.boot.SpringApplication;**

**import org.springframework.boot.autoconfigure.SpringBootApplication;**

**@SpringBootApplication**

**public class SpringLearnApplication {**

**public static void main(String[] args) {**

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

**}**

**}**

**OUTPUT**

****

****