**Create authentication service that returns JWT:**

**Solution:**

JwtAuthServiceApplication.java:

**package** com.cognizant.jwtauthservice;

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

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

@SpringBootApplication

**public** **class** JwtAuthServiceApplication {

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

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

}

}

SecurityConfig.java:

**package** com.cognizant.jwtauthservice.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.web.SecurityFilterChain;

@Configuration

@EnableWebSecurity

**public** **class** SecurityConfig {

@Bean

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

http

.csrf(csrf -> csrf.disable())

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

)

.~~httpBasic~~();

**return** http.build();

}

}

AuthenticationController.java:

**package** com.cognizant.jwtauthservice.controller;

**import** com.cognizant.jwtauthservice.util.JwtUtil;

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

**import** org.springframework.http.\*;

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

**import** java.nio.charset.StandardCharsets;

**import** java.util.Base64;

**import** java.util.Collections;

**import** java.util.Map;

@RestController

**public** **class** AuthenticationController {

@Autowired

**private** JwtUtil jwtUtil;

@GetMapping("/authenticate")

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

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

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

String credentials = **new** String(credDecoded, StandardCharsets.***UTF\_8***);

**final** String[] values = credentials.split(":", 2);

String username = values[0];

String password = values[1];

**if** (!"user".equals(username) || !"pwd".equals(password)) {

**return** ResponseEntity.*status*(HttpStatus.***UNAUTHORIZED***).build();

}

String token = jwtUtil.generateToken(username);

**return** ResponseEntity.*ok*(Collections.*singletonMap*("token", token));

}

}

JwtUtil.java:

**package** com.cognizant.jwtauthservice.util;

**import** io.jsonwebtoken.\*;

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

**import** org.springframework.stereotype.Component;

**import** java.security.Key;

**import** java.util.Date;

@Component

**public** **class** JwtUtil {

**private** **final** String SECRET\_KEY = "myverysecretkeyforjwttesting1234567890";

**private** **final** Key key = Keys.hmacShaKeyFor(SECRET\_KEY.getBytes());

**public** String generateToken(String username) {

**return** Jwts.builder()

.setSubject(username)

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

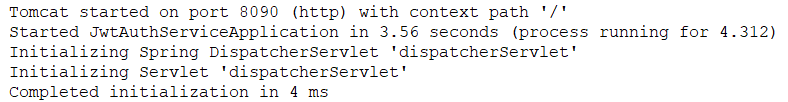
.setExpiration(**new** Date(System.*currentTimeMillis*() + 1000 \* 60 \* 10)) // 10 min expiry

.signWith(key, SignatureAlgorithm.HS256)

.compact();

}

}

OUTPUT: