**Create authentication service that returns JWT**   
package com.cognizant.springlearn.controller;  
  
import com.cognizant.springlearn.util.JwtUtil;  
import jakarta.servlet.http.HttpServletRequest;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.Base64;  
  
@RestController  
public class JwtAuthenticationController {  
  
 private final JwtUtil jwtUtil = new JwtUtil();  
  
 @GetMapping("/authenticate")  
 public ResponseEntity<?> generateToken(HttpServletRequest request) {  
 String authHeader = request.getHeader("Authorization");  
  
 if (authHeader == null || !authHeader.startsWith("Basic ")) {  
 return ResponseEntity.*status*(401).body("Missing or invalid Authorization header");  
 }  
 String base64Credentials = authHeader.substring("Basic ".length()).trim();  
 byte[] decodedBytes = Base64.*getDecoder*().decode(base64Credentials);  
 String credentials = new String(decodedBytes);  
  
 String[] values = credentials.split(":", 2);  
 String username = values[0];  
 String password = values[1];  
 if ("user".equals(username) && "pwd".equals(password)) {  
 String token = jwtUtil.generateToken(username);  
 return ResponseEntity.*ok*("{\"token\":\"" + token + "\"}");  
 } else {  
 return ResponseEntity.*status*(403).body("Invalid credentials");  
 }  
 }  
}

package com.cognizant.springlearn;  
  
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);  
 }  
}

package com.cognizant.springlearn.config;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.Customizer;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.web.SecurityFilterChain;  
  
@Configuration  
public class SecurityConfig {  
  
 @Bean  
 public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
 http  
 .csrf(csrf -> csrf.disable()) // Disable CSRF for testing  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/authenticate").permitAll() // ✅ Allow public access  
 .anyRequest().authenticated() // ⛔ All other endpoints secured  
 )  
 .httpBasic(Customizer.*withDefaults*()); // Enable basic auth  
  
 return http.build();  
 }  
}

package com.cognizant.springlearn.util;  
  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
  
import java.util.Date;  
  
public class JwtUtil {  
 private final String SECRET\_KEY = "my-secret-key";  
  
 public String generateToken(String username) {  
 long now = System.*currentTimeMillis*();  
 return Jwts.*builder*()  
 .setSubject(username)  
 .setIssuedAt(new Date(now))  
 .setExpiration(new Date(now + 3600000)) // 1 hour  
 .signWith(SignatureAlgorithm.*HS256*, SECRET\_KEY)  
 .compact();  
 }  
}

