Authentication Service with JWT in Spring Boot

# Step 1: Add Dependencies in pom.xml

<dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt</artifactId>  
 <version>0.9.1</version>  
</dependency>  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
</dependency>

# Step 2: Create JwtUtil.java for Token Generation

@Component  
public class JwtUtil {  
 private String secret = "mysecretkey";  
  
 public String generateToken(String username) {  
 return Jwts.builder()  
 .setSubject(username)  
 .setIssuedAt(new Date(System.currentTimeMillis()))  
 .setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 10)) // 10 minutes  
 .signWith(SignatureAlgorithm.HS256, secret)  
 .compact();  
 }  
}

# Step 3: Create AuthenticationController.java

@RestController  
public class AuthenticationController {  
  
 @Autowired  
 private JwtUtil jwtUtil;  
  
 @RequestMapping(value = "/authenticate", method = RequestMethod.GET)  
 public ResponseEntity<?> generateToken(HttpServletRequest request) {  
 String authHeader = request.getHeader("Authorization");  
  
 if (authHeader != null && authHeader.startsWith("Basic ")) {  
 String base64Credentials = authHeader.substring("Basic ".length()).trim();  
 byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);  
 String credentials = new String(credDecoded, StandardCharsets.UTF\_8);  
 String[] values = credentials.split(":", 2);  
  
 if (values.length == 2 && "user".equals(values[0]) && "pwd".equals(values[1])) {  
 String token = jwtUtil.generateToken(values[0]);  
 return ResponseEntity.ok(Collections.singletonMap("token", token));  
 }  
 }  
 return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Invalid Credentials");  
 }  
}

# Step 4: Configure SecurityConfig.java

@Configuration  
@EnableWebSecurity  
public class SecurityConfig extends WebSecurityConfigurerAdapter {  
 @Override  
 protected void configure(HttpSecurity http) throws Exception {  
 http.csrf().disable()  
 .authorizeRequests().anyRequest().permitAll();  
 }  
}

# Step 5: Test the Endpoint using Curl

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

{  
 "token": "eyJhbGciOiJIUzI1NiJ9..."  
}

# Final Folder Structure (Optional)

src/  
 ├── controller/  
 │ └── AuthenticationController.java  
 ├── config/  
 │ └── SecurityConfig.java  
 ├── util/  
 │ └── JwtUtil.java