# JWT Authentication Service in Spring Boot

## 1. AuthenticationController.java

package com.example.jwtauth.controller;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.security.authentication.AuthenticationManager;  
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;  
import org.springframework.security.core.Authentication;  
import org.springframework.security.core.AuthenticationException;  
import org.springframework.web.bind.annotation.\*;  
import com.example.jwtauth.util.JwtUtil;  
import java.util.Base64;  
import java.util.HashMap;  
import java.util.Map;  
  
import jakarta.servlet.http.HttpServletRequest;  
  
@RestController  
public class AuthenticationController {  
  
 @Autowired  
 private AuthenticationManager authenticationManager;  
  
 @Autowired  
 private JwtUtil jwtUtil;  
  
 @GetMapping("/authenticate")  
 public Map<String, String> authenticate(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);  
 final String[] values = credentials.split(":", 2);  
  
 try {  
 Authentication auth = authenticationManager.authenticate(  
 new UsernamePasswordAuthenticationToken(values[0], values[1]));  
 String token = jwtUtil.generateToken(auth);  
 Map<String, String> response = new HashMap<>();  
 response.put("token", token);  
 return response;  
 } catch (AuthenticationException e) {  
 throw new RuntimeException("Invalid Credentials");  
 }  
 }  
 throw new RuntimeException("Missing Authorization Header");  
 }  
}

## 2. JwtUtil.java

package com.example.jwtauth.util;  
  
import java.util.Date;  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import org.springframework.security.core.Authentication;  
import org.springframework.stereotype.Component;  
  
@Component  
public class JwtUtil {  
  
 private final String SECRET\_KEY = "secret";  
  
 public String generateToken(Authentication authentication) {  
 String username = authentication.getName();  
 Date now = new Date();  
 Date expiryDate = new Date(now.getTime() + 1000 \* 60 \* 10); // 10 mins  
  
 return Jwts.builder()  
 .setSubject(username)  
 .setIssuedAt(now)  
 .setExpiration(expiryDate)  
 .signWith(SignatureAlgorithm.HS256, SECRET\_KEY)  
 .compact();  
 }  
}

## 3. SecurityConfig.java

package com.example.jwtauth.config;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.authentication.AuthenticationManager;  
import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;  
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();  
 }  
  
 @Bean  
 public AuthenticationManager authenticationManager(AuthenticationConfiguration authenticationConfiguration)  
 throws Exception {  
 return authenticationConfiguration.getAuthenticationManager();  
 }  
}

## 4. Sample Request & Response

Request:

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

Sample Response:

{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5NDc0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-hwKfoqZYlaVQqEUiBloWcWn0ft3tgv0dL0"}