JWT Project 3

This project demonstrates a basic JWT (JSON Web Token) based authentication in Spring Boot. The user sends a username and password and receives a token in return.

# Project Structure

The project is divided into four main classes:

## 1. SecurityConfig.java

Handles security configuration, including in-memory user setup and disabling CSRF.

@Bean  
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
 http.csrf().disable()  
 .authorizeRequests()  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated();  
 return http.build();  
}

## 2. JwtService.java

Generates JWT token with expiration.

public String generateToken(String username) {  
 return Jwts.builder()  
 .setSubject(username)  
 .setIssuedAt(new Date(System.currentTimeMillis()))  
 .setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 60 \* 10))  
 .signWith(SignatureAlgorithm.HS256, SECRET\_KEY)  
 .compact();  
}

## 3. AuthenticationController.java

Handles POST requests at /authenticate. Authenticates and returns a JWT token.

@PostMapping("/authenticate")  
public ResponseEntity<?> authenticate(@RequestParam String username, @RequestParam String password) {  
 try {  
 authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(username, password));  
 String token = jwtService.generateToken(username);  
 return ResponseEntity.ok(Collections.singletonMap("token", token));  
 } catch (Exception e) {  
 return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Invalid credentials");  
 }  
}

## 4. JwtDemoApplication.java

Entry point of the Spring Boot application.

# How to Test Using Postman

Send a POST request to http://localhost:8080/authenticate

Under Body, choose 'x-www-form-urlencoded' and pass:

* - username: user  
  - password: pwd

# Output

{  
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."  
}