**Week 4 : JSON Web Tokens(JWT)**

**Create authentication service that returns JWT**

**Code :**

**SecurityConfig.java**

package com.cogniznt.jwt\_demo.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.config.annotation.web.configuration.EnableWebSecurity;  
import org.springframework.security.config.http.SessionCreationPolicy;  
import org.springframework.security.core.userdetails.User;  
import org.springframework.security.core.userdetails.UserDetailsService;  
import org.springframework.security.core.userdetails.UsernameNotFoundException;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.NoOpPasswordEncoder;  
import org.springframework.security.crypto.password.PasswordEncoder;  
import org.springframework.security.web.SecurityFilterChain;  
  
@Configuration  
@EnableWebSecurity  
public class SecurityConfig {  
  
 @Bean  
 public UserDetailsService userDetailsService() {  
 return username -> {  
 if (username.equals("admin")) {  
 return User.*builder*()  
 .username("admin")  
 .password("admin123") // No encoding with NoOp  
 .roles("USER")  
 .build();  
 } else {  
 throw new UsernameNotFoundException("User not found");  
 }  
 };  
 }  
  
 @Bean  
 public PasswordEncoder passwordEncoder() {  
 return NoOpPasswordEncoder.*getInstance*(); // Only for development/testing  
 }  
  
 @Bean  
 public AuthenticationManager authenticationManager(AuthenticationConfiguration config) throws Exception {  
 return config.getAuthenticationManager();  
 }  
  
 @Bean  
 public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
  
 http  
 .csrf(csrf -> csrf.disable())  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated()  
 )  
 .sessionManagement(session -> session  
 .sessionCreationPolicy(SessionCreationPolicy.*STATELESS*)  
 );  
  
  
 return http.build();  
 }  
  
  
}

**AuthRequest.java**

package com.cogniznt.jwt\_demo.model;  
  
public class AuthRequest {  
 private String username;  
 private String password;  
 // Getters & Setters  
  
 public String getUsername() {  
 return username;  
 }  
  
 public void setUsername(String username) {  
 this.username = username;  
 }  
  
 public String getPassword() {  
 return password;  
 }  
  
 public void setPassword(String password) {  
 this.password = password;  
 }  
}

**AuthController.java**

package com.cogniznt.jwt\_demo.controller;  
  
import com.cogniznt.jwt\_demo.model.AuthRequest;  
import com.cogniznt.jwt\_demo.util.JwtUtil;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.security.authentication.AuthenticationManager;  
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class AuthController {  
  
 @Autowired  
 private AuthenticationManager authenticationManager;  
  
 @Autowired  
 private JwtUtil jwtUtil;  
  
 @PostMapping("/authenticate")  
 public String generateToken(@RequestBody AuthRequest authRequest) throws Exception {  
 System.*out*.println("Received Auth Request: " + authRequest.getUsername());  
  
 try {  
 authenticationManager.authenticate(  
 new UsernamePasswordAuthenticationToken(  
 authRequest.getUsername(), authRequest.getPassword()  
 )  
 );  
 } catch (Exception e) {  
 throw new Exception("Invalid username/password");  
 }  
 return jwtUtil.generateToken(authRequest.getUsername());  
 }  
  
}

**JWTDemoApplicatio.java**

package com.cogniznt.jwt\_demo;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class JwtDemoApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(JwtDemoApplication.class, args);  
 }  
}

**JwtUtil.java**

package com.cogniznt.jwt\_demo.util;  
  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import org.springframework.stereotype.Component;  
  
import java.util.Date;  
  
@Component  
public class JwtUtil {  
  
 private String secret = "secretkey123";  
  
 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)  
 .compact();  
 }  
}

**Output :**

