**WEEK - 4**

**JWT**

**Mandatory**

1. **Create authentication service that returns JWT**

**Security.java**

package com.cognizant.spring\_learn.security;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.web.SecurityFilterChain;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.crypto.password.NoOpPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

import static org.springframework.security.config.Customizer.withDefaults;

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests(auth -> auth

.requestMatchers("/country/").hasRole("USER")

.requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

.anyRequest().authenticated()

)

.httpBasic(withDefaults()) // updated line

.csrf(csrf -> csrf.disable()); // disable CSRF

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

UserDetails user = User.withUsername("user")

.password("pwd")

.roles("USER")

.build();

UserDetails admin = User.withUsername("admin")

.password("adminpwd")

.roles("ADMIN")

.build();

return new InMemoryUserDetailsManager(user, admin);

}

@SuppressWarnings("deprecation")

@Bean

public PasswordEncoder passwordEncoder() {

return NoOpPasswordEncoder.getInstance();

    }

}

