package com.kitchenstory.config;

import org.springframework.beans.factory.annotation.Autowired;

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.builders.AuthenticationManagerBuilder;

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

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

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

import org.springframework.security.config.http.SessionCreationPolicy;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

import com.kitchenstory.services.CustomUserDetailService;

@SuppressWarnings("deprecation")

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter{

@Autowired

private CustomUserDetailService customUserDetailService;

@Autowired

private JwtAuthEntryPoint entryPoint;

@Autowired

JwtAuthFilter authFilter;

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.csrf()

.disable()

.cors().and()

.authorizeRequests()

.antMatchers("/token").permitAll()

.antMatchers("/changePassword").permitAll()

.antMatchers("/getProduct/\*\*").permitAll()

.antMatchers("/getItem/\*\*").permitAll()

.antMatchers("/order").permitAll()

.antMatchers("/getOrder/\*\*").permitAll()

.anyRequest().authenticated()

.and().sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)

.and().exceptionHandling().authenticationEntryPoint(entryPoint);

http.addFilterBefore(authFilter, UsernamePasswordAuthenticationFilter.class);

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(this.customUserDetailService).passwordEncoder(passwordEncoder());

}

@Bean

public BCryptPasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder(10);

}

@Bean

public AuthenticationManager authenticationManagerBean() throws Exception {

return super.authenticationManagerBean();

}

}