package com.simplilearn.capstone2.configuration;

import java.io.IOException;

import javax.servlet.FilterChain;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

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

import org.springframework.context.annotation.Lazy;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.context.SecurityContextHolder;

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

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

import org.springframework.stereotype.Component;

import org.springframework.web.filter.OncePerRequestFilter;

import com.simplilearn.capstone2.service.JwtService;

import com.simplilearn.capstone2.util.JwtUtil;

import io.jsonwebtoken.ExpiredJwtException;

@Component

public class JwtRequestFilter extends OncePerRequestFilter {

public static String CURRENT\_USER = "";

@Autowired

private JwtUtil jwtutil;

@Lazy

@Autowired

private JwtService jwtservice;

@Override

protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response,

FilterChain filterChain)

throws ServletException, IOException {

final String header = request.getHeader("Authorization");

String jwtToken = null;

String userName = null;

if (header != null && header.startsWith("Bearer ")) {

jwtToken = header.substring(7);

try {

userName = jwtutil.getUserNameFromToken(jwtToken);

CURRENT\_USER = userName;

} catch (IllegalArgumentException e) {

System.out.println("Unable to get JWT token");

} catch (ExpiredJwtException e) {

System.out.println("Jwt token is expired");

}

} else {

System.out.println("Jwt token does not start with bearer");

}

if (userName != null && SecurityContextHolder.getContext().getAuthentication() == null) {

UserDetails userDetails = jwtservice.loadUserByUsername(userName);

if (jwtutil.ValidateToken(jwtToken, userDetails)) {

UsernamePasswordAuthenticationToken usernamePasswordAuthenticationToken =

new UsernamePasswordAuthenticationToken(userDetails,

null, userDetails.getAuthorities());

usernamePasswordAuthenticationToken.setDetails

(new WebAuthenticationDetailsSource().buildDetails(request));

SecurityContextHolder.getContext().setAuthentication(usernamePasswordAuthenticationToken);

}

}

filterChain.doFilter(request, response);

}

}

/\*

\* This class wll retrieve the header token

\* once we get the token we validate all deired details and pass the requests\*/