package com.medicare.controller;

import java.security.Principal;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.authentication.BadCredentialsException;

import org.springframework.security.authentication.DisabledException;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

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

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

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.medicare.config.JwtUtil;

import com.medicare.entities.JwtRequest;

import com.medicare.entities.JwtResponse;

import com.medicare.entities.User;

import com.medicare.services.UserDetailService;

@RestController

@CrossOrigin(origins = "\*")

public class JwtController {

@Autowired

private AuthenticationManager authenticationManager;

@Autowired

private UserDetailService userDetailService;

@Autowired

private JwtUtil jwtUtil;

//generate token

@PostMapping("/generate-token")

public ResponseEntity<?> generateToken(@RequestBody JwtRequest jwtRequest) throws Exception{

try {

authenticate(jwtRequest.getUsername(), jwtRequest.getPassword());

}catch(UsernameNotFoundException e) {

e.printStackTrace();

throw new Exception("User does not exist or invalid credentials!");

}

// validated

UserDetails userDetails = this.userDetailService.loadUserByUsername(jwtRequest.getUsername());

String token = this.jwtUtil.generateToken(userDetails);

return ResponseEntity.ok(new JwtResponse(token));

}

private void authenticate(String username, String password) throws Exception {

try {

this.authenticationManager.authenticate(new UsernamePasswordAuthenticationToken(username, password));

} catch (BadCredentialsException e) {

throw new Exception("Invalid Credentials! "+e.getMessage());

}catch(DisabledException e) {

throw new Exception("User Disabled! "+e.getMessage());

}

}

@GetMapping("/current-user")

public User getCurrentUser(Principal principal) {

return (User)this.userDetailService.loadUserByUsername(principal.getName());

}

}