**SpringBoot\_Final Wrap Up**

Updated PostController.java

package org.studyeasy.SpringBlog.controller;

import java.security.Principal;

import java.util.Optional;

import javax.validation.Valid;

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

import org.springframework.security.access.prepost.PreAuthorize;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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

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

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

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

import org.studyeasy.SpringBlog.models.Account;

import org.studyeasy.SpringBlog.models.Post;

import org.studyeasy.SpringBlog.services.AccountService;

import org.studyeasy.SpringBlog.services.PostService;

@Controller

public class PostController {

    @Autowired

    private PostService postService;

    @Autowired

    private AccountService accountService;

    @GetMapping("/posts/{id}")

    public String getPost(@PathVariable Long id, Model model, Principal principal) {

        Optional<Post> optionalPost = postService.getById(id);

        String authUser = "email";

        if (optionalPost.isPresent()) {

            Post post = optionalPost.get();

            model.addAttribute("post", post);

            //get username of current logged in session user

            //String authUsername = SecurityContextHolder.getContext().getAuthentication().getName();

            if (principal != null) {

                authUser = principal.getName();

            }

            if (authUser.equals(post.getAccount().getEmail())){

                model.addAttribute("isOwner", true);

            }else{

                model.addAttribute("isOwner", false);

            }

            return "post\_views/post";

        } else {

            return "404";

        }

    }

    @GetMapping("/posts/add")

    @PreAuthorize("isAuthenticated()")

    public String addPost(Model model, Principal principal) {

        String authUser = "email";

        if(principal != null){

            authUser = principal.getName();

        }

        Optional<Account> optionalAccount = accountService.findOneByEmail(authUser);

        if(optionalAccount.isPresent()){

            Post post = new Post();

            post.setAccount(optionalAccount.get());

            model.addAttribute("post", post);

            return "post\_views/post\_add";

        }else{

            return "redirect:/";

        }

    }

    @PostMapping("/posts/add")

    @PreAuthorize("isAuthenticated()")

    public String addPostHandler(@Valid @ModelAttribute Post post, BindingResult bindingResult ,Principal principal){

        if (bindingResult.hasErrors()){

            return "post\_views/post\_add";

        }

        String authUser = "email";

        if(principal != null){

            authUser = principal.getName();

        }

        if (post.getAccount().getEmail().compareToIgnoreCase(authUser) < 0){

            return "redirect:/?error";

        }

        postService.save(post);

        return "redirect:/posts/"+post.getId();

    }

    @GetMapping("/posts/{id}/edit")

    @PreAuthorize("isAuthenticated()")

    public String getPostForEdit(@PathVariable Long id, Model model){

        Optional<Post> optionaPost = postService.getById(id);

        if(optionaPost.isPresent()){

            Post post = optionaPost.get();

            model.addAttribute("post", post);

            return "post\_views/post\_edit";

        }else{

            return "404";

        }

    }

    @PostMapping("/posts/{id}/edit")

    @PreAuthorize("isAuthenticated()")

    public String updatePost(@Valid @ModelAttribute Post post, BindingResult bindingResult, @PathVariable Long id){

        if (bindingResult.hasErrors()){

            return "post\_views/post\_edit";

        }

        Optional<Post> optionalPost = postService.getById(id);

        if(optionalPost.isPresent()){

            Post existingPost = optionalPost.get();

            existingPost.setTitle(post.getTitle());

            existingPost.setBody(post.getBody());

            postService.save(existingPost);

        }

        return "redirect:/posts/"+post.getId();

    }

    @GetMapping("/posts/{id}/delete")

    @PreAuthorize("isAuthenticated()")

    public String deletePost(@PathVariable Long id){

        Optional<Post> optionalPost = postService.getById(id);

        if(optionalPost.isPresent()){

            Post post = optionalPost.get();

            postService.delete(post);

            return "redirect:/";

        }else{

            return "redirect:/?error";

        }

    }

}