using Identity.Models;

using Microsoft.AspNetCore.Identity;

using Microsoft.AspNetCore.Mvc;

namespace Identity.Controllers

{

public class AdminController : Controller

{

private UserManager<AppUser> userManager;

private IPasswordHasher<AppUser> passwordHasher;

public AdminController(UserManager<AppUser> usrMgr, IPasswordHasher<AppUser> passwordHash)//creating object of generic type

{

userManager = usrMgr;

passwordHasher = passwordHash;

}

public IActionResult Index()

{

return View(userManager.Users);

}

public IActionResult Create() => View();

[HttpPost]

public async Task<IActionResult> Create(User user)

{

if (ModelState.IsValid)

{

AppUser appUser = new AppUser

{

UserName = user.Name,

Email = user.Email,

//TwoFactorEnabled = true

};

IdentityResult result = await userManager.CreateAsync(appUser, user.Password);

/\*if (result.Succeeded)

{

var token = await userManager.GenerateEmailConfirmationTokenAsync(appUser);

var confirmationLink = Url.Action("ConfirmEmail", "Email", new { token, email = user.Email }, Request.Scheme);

EmailHelper emailHelper = new EmailHelper();

bool emailResponse = emailHelper.SendEmail(user.Email, confirmationLink);

if (emailResponse)

return RedirectToAction("Index");

else

{

// log email failed

}

}\*/

if (result.Succeeded)

return RedirectToAction("Index");

else

{

foreach (IdentityError error in result.Errors)

ModelState.AddModelError("", error.Description);

}

}

return View(user);

}

public async Task<IActionResult> Update(string id)

{

AppUser user = await userManager.FindByIdAsync(id);

if (user != null)

return View(user);

else

return RedirectToAction("Index");

}

[HttpPost]

public async Task<IActionResult> Update(string id, string email, string password)

{

AppUser user = await userManager.FindByIdAsync(id);

if (user != null)

{

if (!string.IsNullOrEmpty(email))

user.Email = email;

else

ModelState.AddModelError("", "Email cannot be empty");

if (!string.IsNullOrEmpty(password))

user.PasswordHash = passwordHasher.HashPassword(user, password);

else

ModelState.AddModelError("", "Password cannot be empty");

if (!string.IsNullOrEmpty(email) && !string.IsNullOrEmpty(password))

{

IdentityResult result = await userManager.UpdateAsync(user);

if (result.Succeeded)

return RedirectToAction("Index");

else

Errors(result);

}

}

else

ModelState.AddModelError("", "User Not Found");

return View(user);

}

private void Errors(IdentityResult result)

{

foreach (IdentityError error in result.Errors)

ModelState.AddModelError("", error.Description);

}

[HttpPost]

public async Task<IActionResult> Delete(string id)

{

AppUser user = await userManager.FindByIdAsync(id);

if (user != null)

{

IdentityResult result = await userManager.DeleteAsync(user);

if (result.Succeeded)

return RedirectToAction("Index");

else

Errors(result);

}

else

ModelState.AddModelError("", "User Not Found");

return View("Index", userManager.Users);

}

}

}