**Tạo UserController trong Controller**

**Controller**

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using TechLaptop.Repository;

namespace TechLaptop.Areas.Admin.Controllers

{

[Area("Admin")]

[Authorize (Roles = "Admin")] // Auto phần quyền

public class UserController : Controller

{

private readonly UserManager<AppUserModel> \_userManager;

private readonly RoleManager<IdentityRole> \_roleManager;

private readonly DataContext \_dataContext;

public UserController(UserManager<AppUserModel> userManager , RoleManager<IdentityRole> roleManager, DataContext \_context)

{

\_userManager = userManager;

\_ roleManager = roleManager;

\_dataContext = \_context;

}

public async Task <IActionResult>Index() // hiển thị List AspNetUser

{

// Join 3 Table lại với nhau AspNetUsers vs AspNetUserRoles vs AspNetRoles

var user = await (from u in \_dataContext.Users

join ur in \_dataContext.UserRoles on u.Id equals ur.UserId

join r in \_dataContext.Roles on ur.RoleId equals r.Id

select new {Id = u.Id, UserName = u.UserName,Email = u.Email, PasswordHash =u.PasswordHash, PhoneNumber=u.PhoneNumber, RoleName = r.Name }).ToListAsync();

return View(user);

}

[HttpGet]

public async Task<IActionResult> Create() // Tạo User trong Table AspNetUserRoles

{

var roles = await \_roleManager.Roles.ToListAsync();

ViewBag.Roles = new SelectList(roles, "Id", "Name");

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create(AppUserModel appUser)

{

List<string> errors = new List<string>();

if (ModelState.IsValid)

{

var createUserResult = await \_userManager.CreateAsync(appUser, appUser.PasswordHash);

if (createUserResult.Succeeded)

{

var createUser = await \_userManager.FindByEmailAsync( appUser.Email); // tìm user trong Table bằng AspNetUser bằng Email

var createRole = await \_roleManager.FindByIdAsync(appUser.RoleId); // tìm role trong bảng AspNetRoles bằng RoleId

// tạo 1 bảng ghi AspNetUserRoles liên kết AspNetUser và AspNetRoles

var addToRoleResult = await \_userManager.AddToRoleAsync(createUser, createRole.Name);

// Check nếu thêm role thành công

if (!addToRoleResult.Succeeded)

{

foreach (var error in addToRoleResult.Errors)

{

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

}

}

TempData["success"] = "Tạo thành công";

return RedirectToAction("Index", "User");

}

else

{

foreach (IdentityError error in createUserResult.Errors)

{

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

}

}

}

else

{

TempData["Error"] = "Không thành công";

foreach (var value in ModelState.Values)

{

foreach (var error in value.Errors)

{

errors.Add(error.ErrorMessage);

}

}

string errorMessage = string.Join("\n", errors)

}

return View(appUser);

}

[HttpGet]

public async Task<IActionResult> Delete(string Id) // Delete User trong Table AspNetUser

{

if (string.IsNullOrEmpty(Id))

{

return NotFound();

}

var userId = await \_userManager.FindByIdAsync(Id);

if (userId== null)

{

return NotFound();

}

var result = await \_userManager.DeleteAsync(userId);

if (!result.Succeeded)

{

return View("Error");

}

TempData["success"] = "Xóa Thành Công";

return RedirectToAction ("Index","User");

}

[HttpGet]

public async Task<IActionResult>Edit(string Id) // Edit User trong Table

{

if (string.IsNullOrEmpty(Id))

{

return NotFound();

}

var userId = await \_userManager.FindByIdAsync(Id);

if (userId == null)

{

return NotFound();

}

ViewBag.Roles = new SelectList( await \_roleManager.Roles.ToListAsync(), "Id", "Name", userId.RoleId);

return View(userId);

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit( AppUserModel user)

{

List<string> errors = new List<string>();

var userId = await \_userManager.FindByIdAsync(user.Id);

if (userId == null)

{

return NotFound();

}

if (ModelState.IsValid)

{

userId.UserName = user.UserName;

userId.Email = user.Email;

userId.PhoneNumber = user.PhoneNumber;

userId.PasswordHash = user.PasswordHash;

userId.RoleId = user.RoleId;

var result = await \_userManager.UpdateAsync(userId);

if (result.Succeeded)

{

var createUser = await \_userManager.FindByEmailAsync(user.Email); // tìm user trong Table bằng AspNetUsers bằng Email

var createRole = await \_roleManager.FindByIdAsync(user.RoleId); // tìm role trong bảng AspNetRoles bằng RoleId

// tạo 1 bảng ghi AspNetUserRoles liên kết AspNetUser và AspNetRoles

var addToRoleResult = await \_userManager.AddToRoleAsync(createUser, createRole.Name);

TempData["success"] = "Cập nhật thành công";

return RedirectToAction("Index", "User");

}

else

{

foreach (var error in result.Errors)

{

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

}

}

}

else

{

TempData["Error"] = "Không thành công";

foreach (var value in ModelState.Values)

{

foreach (var error in value.Errors)

{

errors.Add(error.ErrorMessage);

}

}

string errorMessage = string.Join("\n", errors);

}

return View(userId);

}

Models

using Microsoft.AspNetCore.Identity;

namespace TechLaptop.Models

{

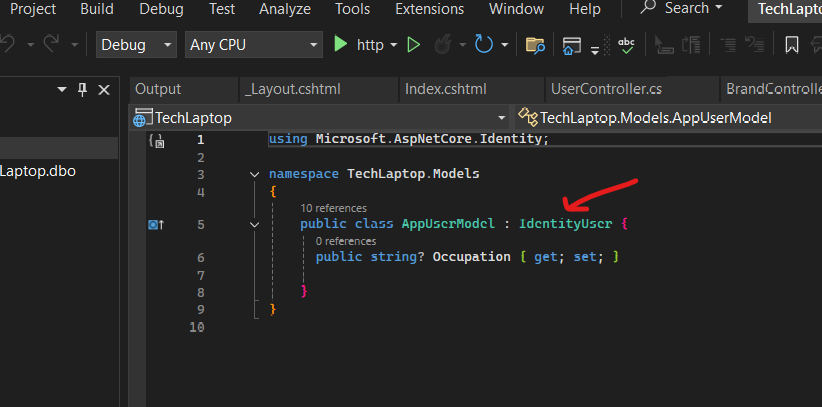
public class AppUserModel : IdentityUser {

public string? Occupation { get; set; }

public string RoleId { get; set; } // để join vs Table AspNetRoles nhưng nhớ bỏ Id của Table AspNetUser và Id của Table AspNetRoles vào trong Table AspNetUserRoles và join 3 Table lại với nhau

}

}



View

Index

@model List<TechLaptop.Models.AppUserModel> // Class trong Models kế thừa IdentityUser

<a asp-action="Create" asp-controller="User">Add User</a>

<table class="table">

<thead>

<tr>

<th scope="col">UserName</th>

<th scope="col">Email</th>

<th scope="col">PasswordHash</th>

<th scope="col">Phone</th>

<th scope="col">Roles</th>

</tr>

</tr>

</thead>

<tbody>

@foreach (var user in Model)

{

<tr>

<td>@user.UserName</td>

<td>@user.Email</td>

<td>@user.PasswordHash</td>

<td>@user.PhoneNumber</td>

<td>@user.RoleName</td>

<td>

<a asp-action="Edit" asp-controller="User" asp-route-Id="@user.Id" class="btn btn-warning btn-sm">Edit</a>

<a asp-action="Delete" asp-controller="User" asp-route-Id="@user.Id" class="btn btn-danger btn-sm confirmDeletion">Delete</a>

</td>

</tr>

}

</tbody>

</table>

**Create**

@model TechLaptop.Models.AppUserModel

<h3> Create Roles </h3>

<div class="col-md-9">

<form asp-action="Create" asp-controller="User" enctype="multipart/form-data">

<div asp-validation-summary="ModelOnly" class="text-danger"> </div>

<div class=" form-group">

<label>UserName</label>

<input asp-for="UserName" class="form-control" />

<span asp-validation-for="UserName" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Email</label>

<textarea asp-for="Email" class="form-control"> </textarea>

<span asp-validation-for="Email" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Phone</label>

<textarea asp-for="PhoneNumber" class="form-control"> </textarea>

<span asp-validation-for="PhoneNumber" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Password</label>

<textarea asp-for="PasswordHash" class="form-control"> </textarea>

<span asp-validation-for="PasswordHash" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Roles</label>

<select asp-for="RoleId" class="form-control" asp-items="ViewBag.Roles"> </select>

<span asp-validation-for="RoleId" class="text-danger"> </span>

</div>

<!--

<div class=" form-group">

<label>Status</label>

<select asp-for="Status" class="form-control">

<option value="">-- Chọn trạng thái --</option>

<option value="1">Hiển thị</option>

<option value="0">Ẩn</option>

</select>

<span asp-validation-for="Status" class="text-danger"> </span>

</div>

-->

<div class=" form-group">

<button class="btn btn-primary">Create</button>

</div>

</form>

<a asp-action="Index" asp-controller="User"> Back to User List</a>

</div>

@section Scripts {

@{

await Html.PartialAsync("\_ValidationScriptsPartial");

}

}

**Edit**

@model TechLaptop.Models.AppUserModel

<h3> Edit Roles </h3>

<div class="col-md-9">

<form asp-action="Edit" asp-controller="User" enctype="multipart/form-data">

<div asp-validation-summary="ModelOnly" class="text-danger"> </div>

<div class=" form-group">

<input asp-for="Id" class="form-control" type="hidden"/>

</div>

<div class=" form-group">

<label>UserName</label>

<input asp-for="UserName" class="form-control" />

<span asp-validation-for="UserName" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Email</label>

<textarea asp-for="Email" class="form-control"> </textarea>

<span asp-validation-for="Email" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Phone</label>

<textarea asp-for="PhoneNumber" class="form-control"> </textarea>

<span asp-validation-for="PhoneNumber" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Password</label>

<textarea asp-for="PasswordHash" class="form-control"> </textarea>

<span asp-validation-for="PasswordHash" class="text-danger"> </span>

</div>

<div class=" form-group">

<label>Roles</label>

<select asp-for="RoleId" class="form-control" asp-items="ViewBag.Roles"> </select>

<span asp-validation-for="RoleId" class="text-danger"> </span>

</div>

<div class=" form-group">

<button class="btn btn-primary">Edit</button>

</div>

</form>

<a asp-action="Index" asp-controller="User"> Back to User List</a>

</div>

@section Scripts {

@{

await Html.PartialAsync("\_ValidationScriptsPartial");

}

}

* Dùng [Authorize] ở Controller Admin

[Area("Admin")]

[Authorize] or [Authorize(Roles = "Admin, Manager")]

* Tức là Phải Login (var result = User.Identity.IsAuthenticated == true ) và có quyền cụ thể mới vào được trang (Controller / action ) Admin
* Create , Delete, Update , Read của Table AspNetRoles

RoleControllers

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Identity;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using System.Threading.Tasks;

using TechLaptop.Models;

using TechLaptop.Repository;

namespace TechLaptop.Areas.Admin.Controllers

{

[Area("Admin")]

[Authorize(Roles = "Admin")]

public class RoleController : Controller

{

private readonly DataContext \_dataContext;

public RoleController(DataContext \_context)

{

\_dataContext = \_context;

}

// Lấy Table AspNetRoles chuyển thành List

public async Task<IActionResult> Index()

{

var roles = await \_dataContext.Roles.OrderByDescending (r => r.Id).ToListAsync();

return View(roles);

}

}

}

[HttpGet]

public async Task<IActionResult> Create()

{

return View();

}

[HttpPost] // Chỉ nhận POST requests

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create(IdentityRole role)

{

// Kiểm tra Role có tồn tại trong AspnetRoles chưa thông qua role.Name

if (!await \_roleManager.RoleExistsAsync(role.Name))

{

var newRole= await \_roleManager.CreateAsync(new IdentityRole(role.Name));

TempData["success"] = "Tạo thành công";

}

return RedirectToAction("Index","Role"); // quay về Controller/Action

}

[HttpGet]

public async Task<IActionResult> Edit(String Id)

{

if (string.IsNullOrEmpty(Id))

{

return NotFound();

}

var roleId = await \_roleManager.FindByIdAsync(Id);

if (roleId == null)

{

return NotFound();

}

return View(roleId);

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(IdentityRole role)

{

if (string.IsNullOrEmpty(role.Id))

{

return NotFound();

}

var roleId = await \_roleManager.FindByIdAsync(role.Id); // lấy bản ghi AspNetRoles theo Id

if (roleId == null)

{

return NotFound();

}

roleId.Name = role.Name;

var result = await \_roleManager.UpdateAsync(roleId);

if (result.Succeeded)

{

TempData["success"] = "Cập nhật thành công";

}

else

{

// Duyệt tất cả lỗi từ Identity và add vào ModelState

foreach (IdentityError error in result.Errors)

{

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

}

}

return RedirectToAction("Index", "Role");

}

public async Task<IActionResult> Delete(string Id) {

var roleId = await \_roleManager.FindByIdAsync(Id);

if (roleId == null )

{

return NotFound();

}

else

{

var result = await \_roleManager.DeleteAsync(roleId);

if (result.Succeeded)

{

TempData["success"] = "Xóa thành công";

return RedirectToAction("Index", "Role");

}

else

{

// 🔹 Duyệt tất cả lỗi từ Identity và add vào ModelState

foreach (IdentityError error in result.Errors)

{

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

}

}

}

return RedirectToAction("Index", "Role");

}

Views

Index

@model List<IdentityRole> // khai báo kiểu dữ liệu của Table AspNetRoles

<**a** **asp-action**="Create" **asp-controller**="Role">Add Role</**a**>

<table class="table">

<thead>

<tr>

<th scope="col"> Id</th>

<th scope="col">Role Name</th>

<th scope="col">Manage Role</th>

</tr>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<th scope="row">@item.Id</th>

<td>@item.Name</td>

<td>

<**a** **asp-action**="Edit" **asp-controller**="Role" **asp-route-Id**="@item.Id" class="btn btn-warning btn-sm">Edit</**a**>

<**a** **asp-action**="Delete" **asp-controller**="Role" **asp-route-Id**="@item.Id" class="btn btn-danger btn-sm confirmDeletion">Delete</**a**>

</td>

</tr>

}

</tbody>

</table>

Create

@model IdentityRole // khai báo kiểu dữ liệu của Table AspNetRoles

<h3> Create Role </h3>

<div class="col-md-9">

<**form** **asp-action**="Create" **asp-controller**="Role" enctype="multipart/form-data">

<**div** **asp-validation-summary**="ModelOnly" class="text-danger"> </**div**>

<div class=" form-group">

<label>Name</label>

<**input** **asp-for**="Name" class="form-control" />

<**span** **asp-validation-for**="Name" class="text-danger"> </**span**>

</div>

<div class=" form-group">

<button class="btn btn-primary">Create</button>

</div>

</**form**>

<**a** **asp-action**="Index" **asp-controller**="Role"> Back to Roles list</**a**>

</div>

@section Scripts {

@{

await Html.PartialAsync("\_ValidationScriptsPartial");

}

}

Edit

@model IdentityRole // khai báo kiểu dữ liệu của Table AspNetRoles

<h3> Edit Roles </h3>

<div class="col-md-9">

<**form** **asp-action**="Edit" **asp-controller**="Role" enctype="multipart/form-data">

<**div** **asp-validation-summary**="ModelOnly" class="text-danger"> </**div**>

<div class=" form-group">

<**input** **asp-for**="Id" class="form-control" **type**="hidden" />

</div>

<div class=" form-group">

<label> Name</label>

<**input** **asp-for**="Name" class="form-control" />

<**span** **asp-validation-for**="Name" class="text-danger"> </**span**>

</div>

<div class=" form-group">

<button class="btn btn-primary">Edit</button>

</div>

</**form**>

<**a** **asp-action**="Index" **asp-controller**="Role"> Back to Roles List</**a**>

</div>

@section Scripts {

@{

await Html.PartialAsync("\_ValidationScriptsPartial");

}

}