### 包： Microsoft.AspNetCore.Authentication.Cookies

### **配置Cookie认证**

在 ASP.NET Core 中，有一个非常重要的依赖注入系统，它贯穿于所有项目中。对于认证系统，同样要先进行注册：

public void ConfigureServices(IServiceCollection services){

services.AddAuthentication(options =>

{

options.DefaultScheme = CookieAuthenticationDefaults.AuthenticationScheme;

})

.AddCookie(options =>

{

// 在这里可以根据需要添加一些Cookie认证相关的配置，在本次示例中使用默认值就可以了。

});

}

****AddCookie**** 用来注册 CookieAuthenticationHandler，由它来完成身份认证的主要逻辑。

public void Configure(IApplicationBuilder app){

app.UseAuthentication();

}

如上，使用UseAuthentication方法注册了AuthenticationMiddleware中间件，它会负责调用对应的Handle。

### **AccountController类**

public class AccountController : Controller {

//A very simplistic user store. This would normally be a database or similar.

public List<ApplicationUser> Users => new List<ApplicationUser>() {

new ApplicationUser { UserName = "darkhelmet", Password = "vespa" },

new ApplicationUser{ UserName = "prezscroob", Password = "12345" }

};

public IActionResult Login(string returnUrl = null) {

TempData["returnUrl"] = returnUrl;

return View();

}

[HttpPost]

public async Task<IActionResult> Login(ApplicationUser user, string returnUrl = null)

{

const string badUserNameOrPasswordMessage = "Username or password is incorrect.";

if (user == null) {

return BadRequest(badUserNameOrPasswordMessage);

}

var lookupUser = Users.FirstOrDefault(u => u.UserName == user.UserName);

if (lookupUser?.Password != user.Password) {

return BadRequest(badUserNameOrPasswordMessage);

}

var identity = new ClaimsIdentity(CookieAuthenticationDefaults.AuthenticationScheme);

identity.AddClaim(new Claim(ClaimTypes.Name, lookupUser.UserName));

await HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme, new ClaimsPrincipal(identity));

if(returnUrl == null) {

returnUrl = TempData["returnUrl"]?.ToString();

}

if(returnUrl != null) {

return Redirect(returnUrl);

}

return RedirectToAction(nameof(HomeController.Index), "Home");

}

public async Task<IActionResult> Logout() {

await HttpContext.SignOutAsync(CookieAuthenticationDefaults.AuthenticationScheme);

return RedirectToAction(nameof(HomeController.Index), "Home");

}

}

### **运行**

请求：

GET http://localhost:5000/profile HTTP/1.1

Host: localhost:5000

响应：

HTTP/1.1 302 Found

Location: http://localhost:5000/Account/Login?ReturnUrl=%2Fprofile

因为我们没有登录，会在授权中间件中会执行context.ChallengeAsync();方法，最终会跳转到登录页面，然后输入用户名密码(alice/alice)，登录成功：

请求：

POST http://localhost:5000/Account/Login?ReturnUrl=%2Fprofile HTTP/1.1

Host: localhost:5000

Content-Type: application/x-www-form-urlencoded

returnUrl=%2Fprofile&userName=alice&password=alice

响应：

HTTP/1.1 302 Found

Location: /profile

Set-Cookie: .AspNetCore.Cookies=CfDJ8B4XRZETkRhMt3mT9VduB8KTzFhbcuszdNDXZpaQI3zSOcOD8uAzjr-iHzNCPVgXrKqxfK-MrP5d5r9X1zfKOgg2\_j54t0ccAQ5nshSmXnRvjIZ6id3GD5fDP9v2x1iV0JE7X9IdoA458DZjx6qm6971GeY5HYVnT7odwgQR8eRaHo0-Wacmt95QuC9IVSapqsShHOeu5ZowFmDAPXrlUHOSwBPAjiLkf8mNbu8U4ZcWFlaBXC9-H-2\_ts5wyi-90zw6jGxX3o7tRiQB4qq8IDmIJbZtN4Nl8TKHHcTbyFl5Z\_\_MrgrjJ7s4cGdnIoDJWB9ENw1IGRgF3Rib8KmhkwhlUyO2VMnuVI8vSP2PcwrkUGtudJwHMHrA8cuS021xpmIhkhgW3e82r\_0\_jxAh1nqG4zwTP5i8iLU6FsOLLWatveSWB441Ntqw-L-pYczsBAYFRT0Hh56ofUAxGd7aaGtDx0jvuuxW5gK245Pf0TKG-4G46yDwLrFtjNcN\_GREbpwtHAz-I7XqiDZgS3nbzjik5s05NxB7d6X3aOFc5JHCwFxW-i-xW-ToJLZrp3Jo8W0bAxVwxZIW2PwZlVtyeYSkqByFRaDS4qcBywE2Bmar\_TyJm9UpVWaL2s9KxpU\_DHN6meYne5E5dH4-k1DoABl6FyNPn6xYfMWxzu0\_7ZFhVJjCycScy1jggCv4Hk5nkltj9A3QrFpNb\_HCk21Uek9g-7Zi150EKfDzhGjMto5\_hbWcmQtUsHuLbZlnYTHXZ-7zELZOepAUts2ZGoUnEaI; path=/; samesite=lax; httponly

最后，点击 “退出” ，响应报文中会将Cookie设置为空（清除Cookie）:

请求：

GET http://localhost:5000/Account/Logout HTTP/1.1

Host: localhost:5000

响应：

HTTP/1.1 302 Found

Location: /

Set-Cookie: .AspNetCore.Cookies=; expires=Thu, 01 Jan 1970 00:00:00 GMT; path=/; samesite=lax