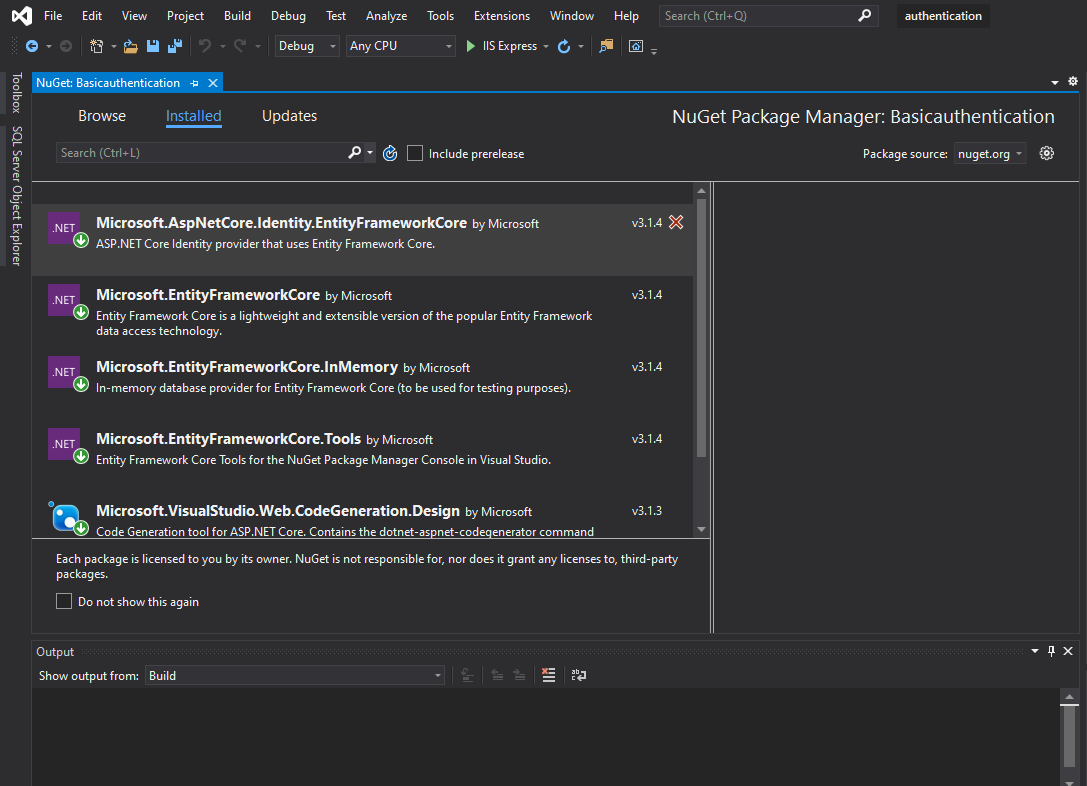
Authenticcation（一）

1. 在nuget包中引入Microsoft.AspNetCore.Idnetity.EntityFrameworkCore

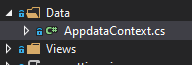
Microsoft..EntityFrameworkCore

Microsoft..EntityFrameworkCore.InMemory



1. 数据源

在根目录下创建Data目录



类中的代码

|  |
| --- |
| public class AppdataContext:IdentityDbContext  {  public AppdataContext(DbContextOptions<AppdataContext> options):base(options)  {  }  } |

IdentityDbContext：Entity Framework数据库上下文用于身份识别的基类。

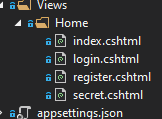
1. Startup配置文件

|  |
| --- |
| public class Startup  {  public void ConfigureServices(IServiceCollection services)  {  //内存数据库  services.AddDbContext<AppdataContext>(config=> {  config.UseInMemoryDatabase("Memory");  });  //addidentity注册服务 密码配置  services.AddIdentity<IdentityUser, IdentityRole>(config=> {  config.Password.RequiredLength = 2;//长度  config.Password.RequireDigit = false;//数字  config.Password.RequireNonAlphanumeric = false;  config.Password.RequireUppercase = false;  })  .AddEntityFrameworkStores<AppdataContext>()  .AddDefaultTokenProviders();  services.ConfigureApplicationCookie(config=> {  config.Cookie.Name = "tmc";  config.LoginPath = "/home/login";  });  services.AddControllersWithViews();    }  // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.  public void Configure(IApplicationBuilder app, IWebHostEnvironment env)  {  if (env.IsDevelopment())  {  app.UseDeveloperExceptionPage();  }  app.UseRouting();  app.UseAuthentication();  app.UseAuthorization();  app.UseEndpoints(endpoints =>  {  endpoints.MapDefaultControllerRoute();  });  }  } |

1. HomeController

|  |
| --- |
| 1. public class HomeController:Controller 2. { 3. //UserManager包含在Microsoft.AspNetCore.Identity中 4. private UserManager<IdentityUser> \_userManager; 5. // SignInManager用于判断用户登录、登出等 6. private SignInManager<IdentityUser> \_signinManager; 7. public HomeController(UserManager<IdentityUser> userManager, 8. SignInManager<IdentityUser> signInManager) 9. { 10. \_userManager = userManager; 11. \_signinManager = signInManager; 12. } 13. public IActionResult index() 14. { 15. return View(); 16. } 17. [Authorize] 18. public IActionResult secret() 19. { 20. return View(); 21. } 22. public IActionResult login() 23. { 25. return View(); 26. } 27. public IActionResult register() 28. { 29. return View(); 30. } 31. [HttpPost] 32. public async Task<IActionResult> login(string um,string pss) 33. { 34. var user =await \_userManager.FindByNameAsync(um); 35. if (user != null) 36. { 37. //登录 38. var signinresult= await \_signinManager.PasswordSignInAsync(user,pss,false,false); 39. if (signinresult.Succeeded) 40. { 41. Redirect("secret"); 42. } 43. } 44. return Redirect("index"); 45. } 46. [HttpPost] 47. public async Task<IActionResult> register(string um, string pss) 48. { 49. var user = new IdentityUser 50. { 51. UserName = um, 52. Email="" 53. }; 54. var result = await \_userManager.CreateAsync(user,pss); 55. Console.WriteLine(result.Succeeded); 56. foreach(var i in result.Errors) 57. { 58. Console.WriteLine(i.Description); 59. } 61. if (result.Succeeded) 62. { 63. //sign here 64. var signinresult = await \_signinManager.PasswordSignInAsync(user, pss, false, false); 65. if (signinresult.Succeeded) 66. { 67. return Redirect("secret"); 68. } 69. } 71. return Redirect("index"); 72. } 73. public async Task<IActionResult> logout() 74. { 75. await \_signinManager.SignOutAsync(); 76. return Redirect("index"); 77. } 78. } |

5. 根据action添加视图



6. index和secret中保持默认

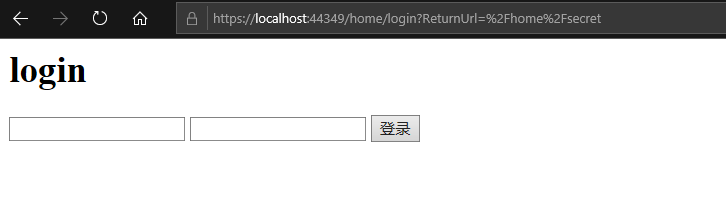
|  |
| --- |
| <h1>login</h1>  <form action="/home/login" method="post">  <input type="text" name="um" value="" />  <input type="text" name="pss" value="" />  <input type="submit" name="n" value="登录" />  </form> |

|  |
| --- |
| <h1>register</h1>  <form action="/home/register" method="post">  <input type="text" name="um" value="" />  <input type="text" name="pss" value="" />  <input type="submit" name="n" value="注册" />  </form> |

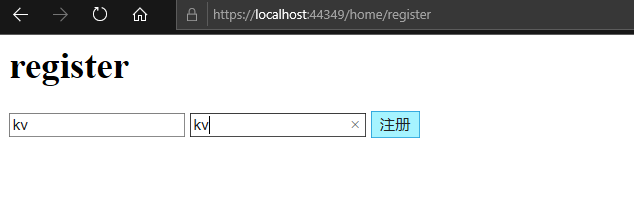
7. 运行结果



未登录状态跳转到



注册



重新登录

能访问到secret页面。