<https://github.com/Anamika-s/jwt-authentication.git>

Add class in Models folder

Add Login Controller with 2 actions methods , Login

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Configuration;

using MvcClient.Models;

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

namespace MvcClient.Controllers

{

public class LoginController : Controller

{

IConfiguration \_config;

public LoginController(IConfiguration configuration)

{

\_config = configuration;

}

public class JWT

{

public string Token { get; set; }

}

[HttpGet]

public IActionResult Login()

{

UserViewModel user = new UserViewModel();

return View(user);

}

[HttpPost]

public IActionResult Login(UserViewModel user)

{

HttpClient client = new HttpClient();

var token = string.Empty;

StringContent content = new StringContent(JsonConvert.SerializeObject(user), Encoding.UTF8, "application/json");

string endpoint = "https://localhost:44339/api/Authentication";

client.BaseAddress = new Uri(endpoint);

var contentType = new MediaTypeWithQualityHeaderValue

("application/json");

client.DefaultRequestHeaders.Accept.Add(contentType);

var Response = client.PostAsync(endpoint, content);

var result = Response.Result;

if (result.IsSuccessStatusCode)

{

var stringJWT = result.Content.ReadAsStringAsync().Result;

JWT jwt = JsonConvert.DeserializeObject

<JWT>(stringJWT);

HttpContext.Session.SetString("token", jwt.Token);

ViewBag.Message = "User logged in successfully!";

//HttpContext.Session["token"] = token.ToString();

//return View();

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

}

else

return View();

}

}

}

public class JWT

{

public string Token { get; set; }

}

StudentController

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.Logging;

using MvcClient.Models;

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Threading.Tasks;

namespace MvcClient.Controllers

{

public class StudentController : Controller

{

private readonly ILogger<StudentController> \_logger;

private readonly IHttpContextAccessor \_httpContextAccessor;

private readonly ISession \_session;

private IConfiguration \_Configure { get; set; }

string apiBaseUrl = "";

public StudentController(ILogger<StudentController> logger, IConfiguration configuration, IHttpContextAccessor httpContextAccessor)

{

\_httpContextAccessor = httpContextAccessor;

\_session = \_httpContextAccessor.HttpContext.Session;

\_logger = logger;

\_Configure = configuration;

apiBaseUrl = \_Configure.GetValue<string>("WebAPIBaseUrl");

}

public async Task<IActionResult> Index()

{

if (TempData["msg"] != null)

ViewBag.msg = "<script> alert('Record Inserted'); </script>";

List<Student> students = new List<Student>();

using (var client = new HttpClient())

{

//Send HTTP requests from here.

// string endpoint = "https://localhost:44339/api/";

client.DefaultRequestHeaders.Clear();

var contentType = new MediaTypeWithQualityHeaderValue

("application/json");

client.DefaultRequestHeaders.Accept.Add(contentType);

client.DefaultRequestHeaders.Authorization =

new AuthenticationHeaderValue("Bearer",

HttpContext.Session.GetString("token"));

var request = new HttpRequestMessage(HttpMethod.Get, "https://localhost:44339/api/students1");

HttpResponseMessage response = await client.SendAsync(request);

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync().Result;

students = JsonConvert.DeserializeObject<List<Student>>(jsonString);

}

return View(students);

}

}

}

}

Start Up File

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.HttpsPolicy;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.IdentityModel.Tokens;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MvcClient

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("ThisismySecretKey"));

services.AddSession(options =>

{

options.IdleTimeout = TimeSpan.FromMinutes(15);

});

services.AddControllersWithViews();

services.AddAuthentication(options =>

{

options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;

options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

}).AddJwtBearer(o => o.TokenValidationParameters = new Microsoft.IdentityModel.Tokens.TokenValidationParameters()

{

ValidateIssuerSigningKey = true,

IssuerSigningKey = key,

ValidateIssuer = true,

ValidIssuer = "Test.com",

ValidateAudience = true,

ValidAudience = "Test.com"

});

services.AddSingleton<IHttpContextAccessor, HttpContextAccessor>();

services.AddMemoryCache();

}

// 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();

}

else

{

app.UseExceptionHandler("/Home/Error");

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseSession();

app.UseAuthentication();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

});

}

}

}

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"AllowedHosts": "\*",

"WebAPIBaseUrl": "https://localhost:44339/api/"

}