using CourseApi.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Newtonsoft.Json;

using System.Net.Http.Headers;

using System.Text;

namespace ClientApp.Controllers

{

public class CourseController : Controller

{

// GET: CourseController

public async Task<ActionResult> Index()

{

string ErrorMsg = string.Empty;

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

HttpResponseMessage response = await client.GetAsync("api/Student");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

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

return View(student);

}

else

{

ErrorMsg = response.ReasonPhrase;

ViewBag.msg = ErrorMsg;

return View();

}

}

}

// GET: CourseController/Details/5

public async Task<ActionResult> Details(int id)

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

HttpResponseMessage response = await client.GetAsync("api/Student/id");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return View(student);

}

else

{

Console.WriteLine(response.ReasonPhrase);

Console.WriteLine("Internal server Error");

}

}

return View();

}

// GET: CourseController/Create

public ActionResult Create()

{

return View();

}

// POST: CourseController/Create

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<ActionResult> Create(Student collection)

{

try

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

JsonConvert.SerializeObject(collection);

var requestContent = new StringContent(JsonConvert.SerializeObject(collection), Encoding.UTF8, "application/json");

HttpResponseMessage response = await client.PostAsync("api/Student", requestContent);

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return RedirectToAction(nameof(Index));

}

else

return View();

}

}

catch

{

return View();

}

}

// GET: CourseController/Edit/5

public async Task<ActionResult> Edit(int id)

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

HttpResponseMessage response = await client.GetAsync("api/Student/id");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return View(student);

}

else

{

Console.WriteLine(response.ReasonPhrase);

Console.WriteLine("Internal server Error");

return View();

}

}

}

// POST: CourseController/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<ActionResult> Edit(int id, Student collection)

{

try

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

JsonConvert.SerializeObject(collection);

var requestContent = new StringContent(JsonConvert.SerializeObject(collection), Encoding.UTF8, "application/json");

HttpResponseMessage response = await client.PutAsync("api/Student/id", requestContent);

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return RedirectToAction(nameof(Index));

}

else

return View();

}

}

catch

{

return View();

}

}

public async Task<ActionResult> Delete(int id)

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

HttpResponseMessage response = await client.GetAsync("api/Student/id");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return View(student);

}

else

{

Console.WriteLine(response.ReasonPhrase);

Console.WriteLine("Internal server Error");

return View();

}

}

}

// POST: CourseController/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<ActionResult> Deleted(int id)

{

try

{

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

//GET Method

//var requestContent = new StringContent(JsonConvert.SerializeObject(collection), Encoding.UTF8, "application/json");

HttpResponseMessage response = await client.DeleteAsync("api/Student/id");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

Student student = JsonConvert.DeserializeObject<Student>(jsonString.Result);

return RedirectToAction(nameof(Index));

}

else

return View();

}

}

catch

{

return View();

}

}

}

}

LoginController

public class LoginController : Controller

{

HttpClient \_httpClient;

//private readonly HttpClient \_httpClient;

//public LoginController(HttpClient httpClient)

//{

// \_httpClient = httpClient;

//}

public async Task<ActionResult> Login()

{

return View();

}

public class JWT

{

public string Token { get; set; }

}

[HttpPost]

public async Task<ActionResult> Login(User user)

{

string baseUrl = "http://localhost:5164";

HttpClient client = new HttpClient();

client.BaseAddress = new Uri(baseUrl);

var contentType = new MediaTypeWithQualityHeaderValue

("application/json");

client.DefaultRequestHeaders.Accept.Add(contentType);

string stringData = JsonConvert.SerializeObject(user);

var contentData = new StringContent(stringData,

System.Text.Encoding.UTF8, "application/json");

HttpResponseMessage response = client.PostAsync

("/api/login", contentData).Result;

string stringJWT = response.Content.

ReadAsStringAsync().Result;

JWT jwt = JsonConvert.DeserializeObject

<JWT>(stringJWT);

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

ViewBag.Message = HttpContext.Session.GetString("token");

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

return View();

}

}

}

After passing token

public async Task<ActionResult> Index()

{

string ErrorMsg = string.Empty;

using (var client = new HttpClient())

{

//Send HTTP requests from here.

client.BaseAddress = new Uri("http://localhost:5164/");

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

string token = HttpContext.Session.GetString("token").ToString();

if(token!=null)

client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", token);

//GET Method

HttpResponseMessage response = await client.GetAsync("api/Student");

if (response.IsSuccessStatusCode)

{

var jsonString = response.Content.ReadAsStringAsync();

jsonString.Wait();

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

return View(student);

}

else

{

ErrorMsg = response.ReasonPhrase;

ViewBag.msg = ErrorMsg;

return View();

}

}

}