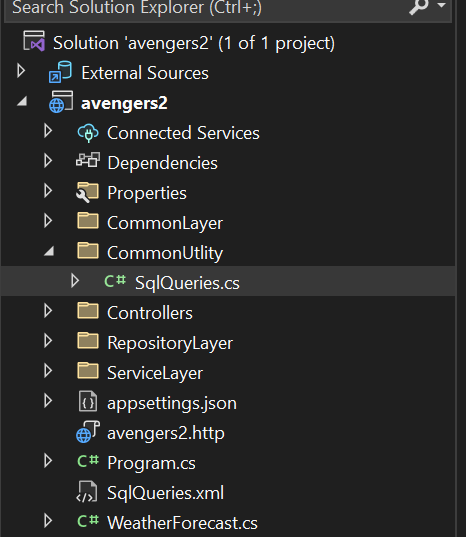
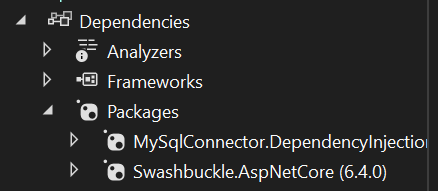
  
<https://www.youtube.com/watch?v=AH1-W4dj32U&list=PLwxiRZdZ4bZmnNjb7M3byWHlGr0mIG_mH&index=6>



**Dependencies**:

Sql and swagger extension



**Properties**> **launchSetting.json**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

{

"$schema": "http://json.schemastore.org/launchsettings.json",

"iisSettings": {

"windowsAuthentication": false,

"anonymousAuthentication": true,

"iisExpress": {

"applicationUrl": "http://localhost:63019",

"sslPort": 44396

}

},

"profiles": {

"http": {

"commandName": "Project",

"dotnetRunMessages": true,

"launchBrowser": true,

"launchUrl": "swagger",

"applicationUrl": "http://localhost:5006",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

},

"https": {

"commandName": "Project",

"dotnetRunMessages": true,

"launchBrowser": true,

"launchUrl": "swagger",

"applicationUrl": "https://localhost:7231;http://localhost:5006",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

},

"IIS Express": {

"commandName": "IISExpress",

"launchBrowser": true,

"launchUrl": "swagger",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

}

}

}

**CommandLayer => Model >Readallinformation.cs**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

namespace avengers2.CommonLayer.Model

{

public class ReadAllInformationResponse

{

public bool IsSuccess { get; set; }

public string? Message { get; set; }

public List<GetReadAllInformation>? readAllInformation { get; set; }

}

public class GetReadAllInformation

{

public int id { get; set; }

public string? UserName { get; set; }

public string? EmailID { get; set; }

public string? MobileNumber { get; set; }

}

}

**CommonUtility => SQLQueries.cs**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.Extensions.Configuration;

namespace avengers2.CommonUtlity

{

public class SqlQueries

{

public static IConfiguration \_configuration = new ConfigurationBuilder().AddXmlFile("SqlQueries.xml",true,true).Build();

public static string AddInformation { get { return \_configuration["AddInformation"]; } }

public static string ReadAllInformation { get { return \_configuration["ReadAllInformation"]; } }

}

}

**Controller => CrudApplicationCOntroller.cs**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

using avengers2.CommonLayer.Model;

using avengers2.ServiceLayer;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace avengers2.Controllers

{

[Route("api/[controller]/[action]")]

[ApiController]

public class CrudApplicationController : ControllerBase

{

public readonly ICRudApplicationSL \_cRudApplicationSL;

public CrudApplicationController(ICRudApplicationSL cRudApplicationSL) {

\_cRudApplicationSL = cRudApplicationSL;

}

[HttpPost]

public async Task<IActionResult> AddInformation(ViewUserInformationRequest request) {

ViewUserInformationResponse response = new ViewUserInformationResponse();

try {

response = await \_cRudApplicationSL.AddInformation(request);

}

catch (Exception ex) {

response.IsSuccess = false;

response.Message = ex.Message;

}

return Ok(response);

}

[HttpGet]

public async Task<IActionResult> ReadAllInformation()

{

ReadAllInformationResponse response = new ReadAllInformationResponse();

try

{

response = await \_cRudApplicationSL.ReadAllInformation();

if (!response.IsSuccess) {

return BadRequest(new { IsSuccess = response.IsSuccess, Message = response.Message , Data=response.readAllInformation});

}

}

catch (Exception ex)

{

response.IsSuccess = false;

response.Message = ex.Message;

return BadRequest(new { IsSuccess = response.IsSuccess, Message = response.Message });

}

return Ok(new { IsSuccess = response.IsSuccess, Message = response.Message, Data = response.readAllInformation });

}

}

}

**RepositryLayer =>**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

using avengers2.CommonLayer.Model;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.Extensions.Configuration;

using MySqlConnector;

using avengers2.CommonUtlity;

using System.Runtime.InteropServices;

namespace avengers2.RepositoryLayer

{

public class CrudApplicationRL : ICRudApplicationRL

{

public readonly IConfiguration \_configuration;

public readonly MySqlConnection \_mySqlConnection;

public CrudApplicationRL(IConfiguration configuration) {

\_configuration = configuration;

\_mySqlConnection = new MySqlConnection(\_configuration["ConnectionStrings:Default"]);

}

public async Task<ViewUserInformationResponse> AddInformation(ViewUserInformationRequest request)

{

ViewUserInformationResponse response = new ViewUserInformationResponse();

response.IsSuccess = true;

response.Message = "Success";

try

{

if (\_mySqlConnection.State != System.Data.ConnectionState.Open)

{

await \_mySqlConnection.OpenAsync();

}

using (MySqlCommand sqlCommand = new MySqlCommand(SqlQueries.AddInformation, \_mySqlConnection))

{

sqlCommand.CommandType = System.Data.CommandType.Text;

sqlCommand.CommandTimeout = 180;

sqlCommand.Parameters.AddWithValue("@UserName", request.UserName);

sqlCommand.Parameters.AddWithValue("@EmailID", request.EmailID);

sqlCommand.Parameters.AddWithValue("@MobileNumber", request.MobileNumber);

int Status = await sqlCommand.ExecuteNonQueryAsync();

if (Status <= 0)

{

response.IsSuccess = false;

response.Message = "Query Not Executed";

return response;

}

}

}

catch (Exception ex)

{

response.IsSuccess = false;

response.Message = ex.Message;

}

finally {

await \_mySqlConnection.CloseAsync();

await \_mySqlConnection.DisposeAsync();

}

return response;

}

public async Task<ReadAllInformationResponse> ReadAllInformation()

{

ReadAllInformationResponse response = new ReadAllInformationResponse();

response.IsSuccess = true;

response.Message = "Success";

try {

if (\_mySqlConnection.State != System.Data.ConnectionState.Open)

{

await \_mySqlConnection.OpenAsync();

}

using (MySqlCommand sqlCommand = new MySqlCommand(SqlQueries.ReadAllInformation, \_mySqlConnection))

{

try {

sqlCommand.CommandType = System.Data.CommandType.Text;

sqlCommand.CommandTimeout = 180;

using ( MySqlDataReader dataReader = await sqlCommand.ExecuteReaderAsync())

{

if (dataReader.HasRows)

{

response.readAllInformation = new List<GetReadAllInformation>();

while (await dataReader.ReadAsync())

{

GetReadAllInformation getdata = new GetReadAllInformation();

getdata.id = dataReader["id"] != DBNull.Value ? Convert.ToInt32(dataReader["id"]) : 0;

getdata.UserName = dataReader["UserName"] != DBNull.Value ? Convert.ToString(dataReader["UserName"]) : String.Empty;

getdata.EmailID = dataReader["EmailID"] != DBNull.Value ? Convert.ToString(dataReader["EmailID"]) : String.Empty;

getdata.MobileNumber = dataReader["MobileNumber"] != DBNull.Value ? Convert.ToString(dataReader["MobileNumber"]) : String.Empty;

response.readAllInformation.Add(getdata);

}

}

else {

response.IsSuccess = false;

response.Message = "Record Not Found";

}

}

}

catch (Exception ex) {

response.IsSuccess = false;

response.Message = ex.Message;

}

finally {

await sqlCommand.DisposeAsync();

}

}

}

catch (Exception ex) {

response.IsSuccess = false;

response.Message = ex.Message;

}

finally

{

await \_mySqlConnection.CloseAsync();

await \_mySqlConnection.DisposeAsync();

}

return response;

}

}

}

**serviceLayer**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

using avengers2.CommonLayer.Model;

using avengers2.RepositoryLayer;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace avengers2.ServiceLayer

{

public class CrudApplicationSL : ICRudApplicationSL

{

public readonly ICRudApplicationRL \_cRudApplicationRL;

public CrudApplicationSL(ICRudApplicationRL cRudApplicationRL) {

\_cRudApplicationRL = cRudApplicationRL;

}

public async Task<ViewUserInformationResponse> AddInformation(ViewUserInformationRequest request)

{

ViewUserInformationResponse response = new ViewUserInformationResponse();

if (String.IsNullOrEmpty(request.UserName)) {

response.IsSuccess = false;

response.Message = "Username Can not null";

return response;

}

return await \_cRudApplicationRL.AddInformation(request);

}

public async Task<ReadAllInformationResponse> ReadAllInformation() {

return await \_cRudApplicationRL.ReadAllInformation();

}

}

}

**Appsetting.json**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"ConnectionStrings": {

"Default": "server=localhost;user=root;password='';database=thar;port=3306;SslMode=none;Allow User Variables=true;"

},

"AllowedHosts": "\*"

}

**Avengers.http**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

@avengers2\_HostAddress = http://localhost:5006

GET {{avengers2\_HostAddress}}/weatherforecast/

Accept: application/json

###

**Program.cs**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

using avengers2.RepositoryLayer;

using avengers2.ServiceLayer;

using MySqlConnector;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

builder.Services.AddScoped<ICRudApplicationSL, CrudApplicationSL>();

builder.Services.AddScoped<ICRudApplicationRL, CrudApplicationRL>();

// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddMySqlDataSource(builder.Configuration.GetConnectionString("Default")!);

var app = builder.Build();

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

**SQLqueries.xml**

**==========9999============333==========44444444444444444=55555555555555555555=**

**==========9999============333==========44444444444444444=55555555555555555555=**

<?xml version="1.0" encoding="utf-8" ?>

<Queries>

<AddInformation>

<![CDATA[

INSERT INTO user ( UserName, MobileNumber, EmailID) VALUES ( @UserName, @MobileNumber, @EmailID);

]]>

</AddInformation>

<ReadAllInformation>

<![CDATA[

select \* from user

]]>

</ReadAllInformation>

</Queries>