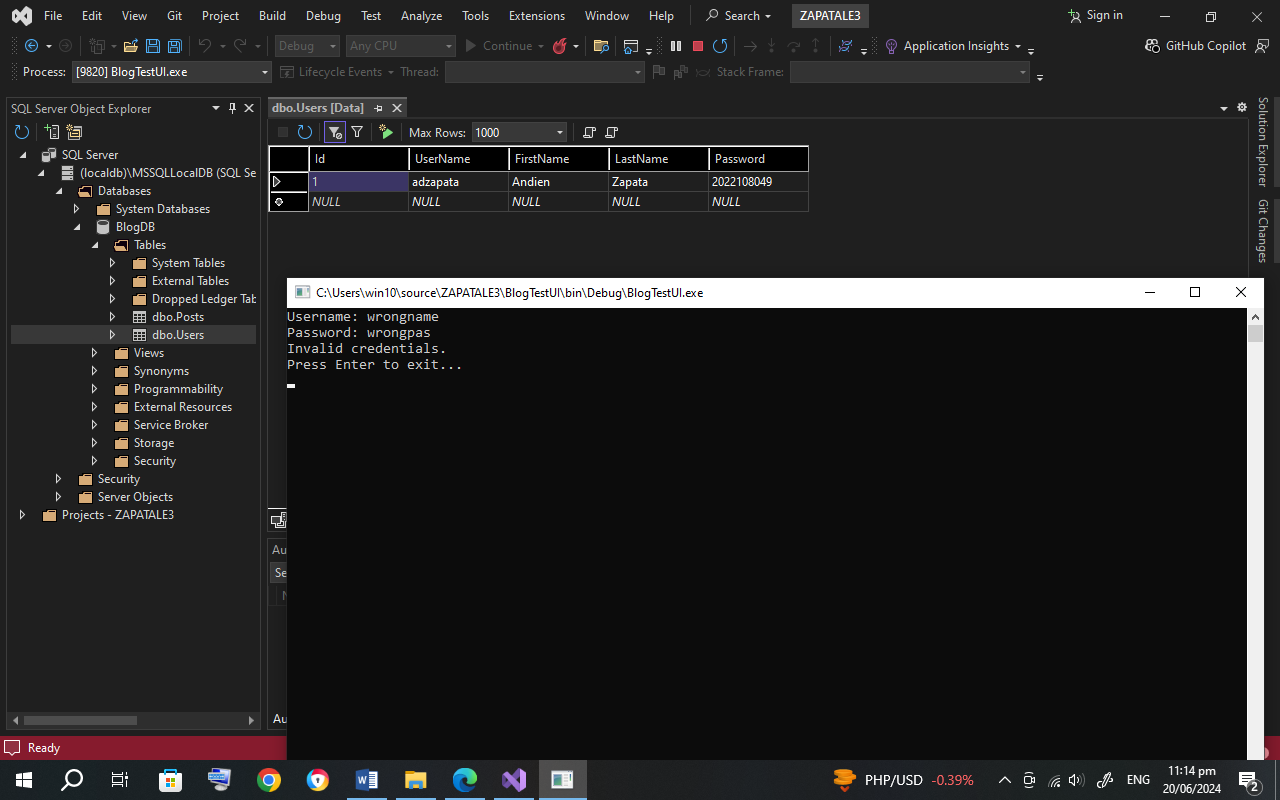
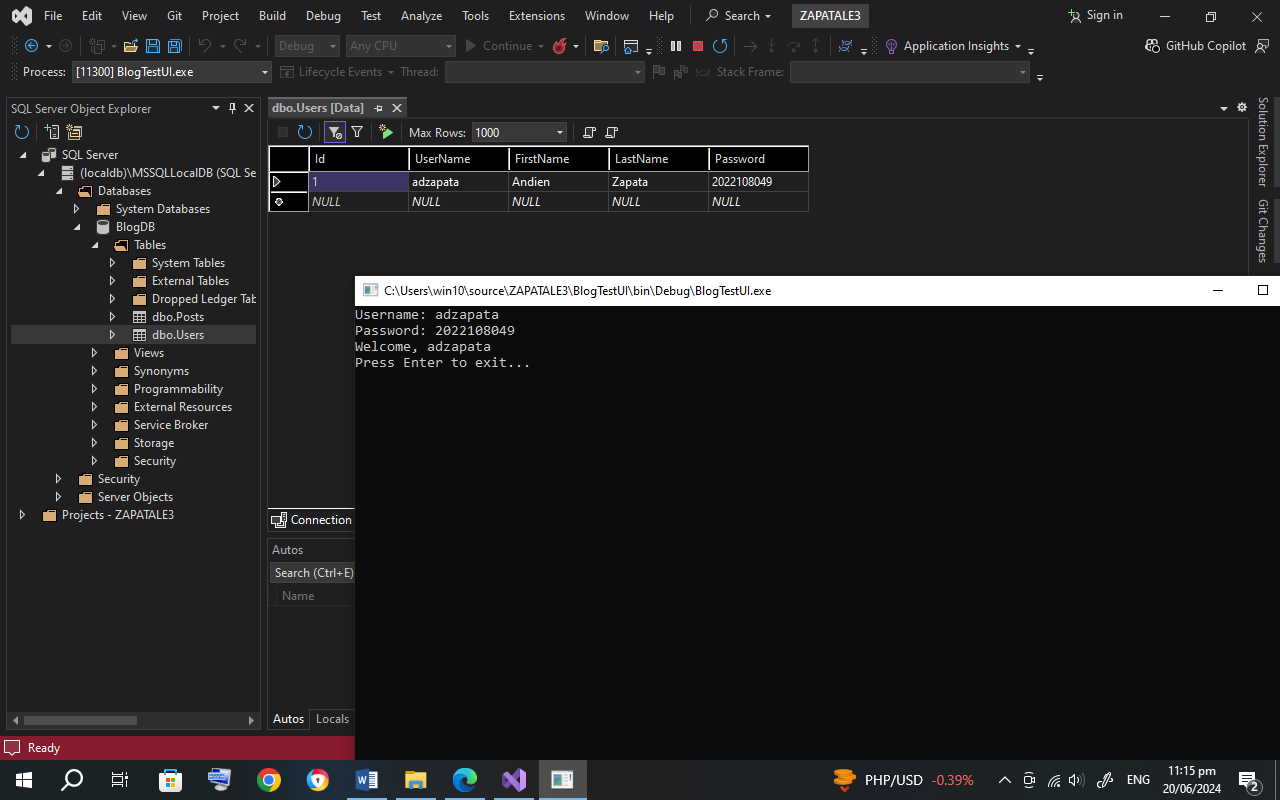
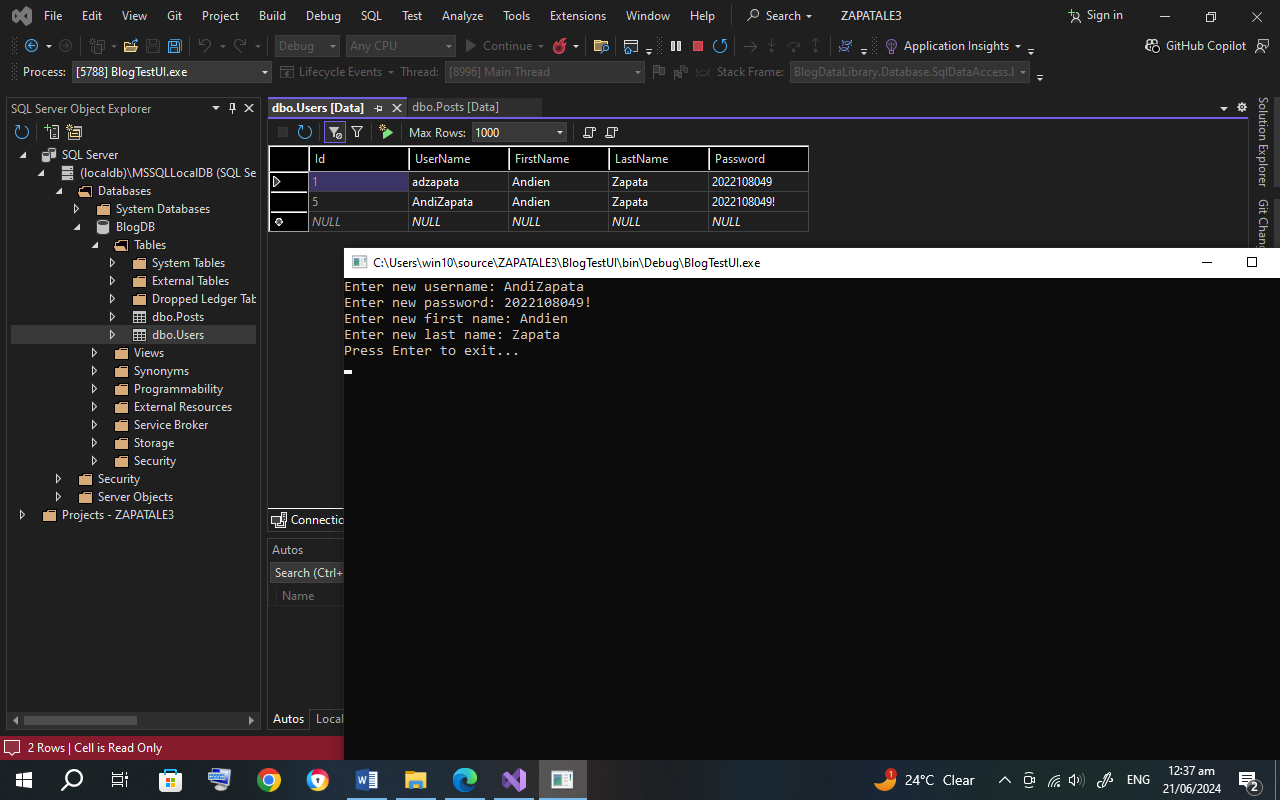
# App functionality: Login Test

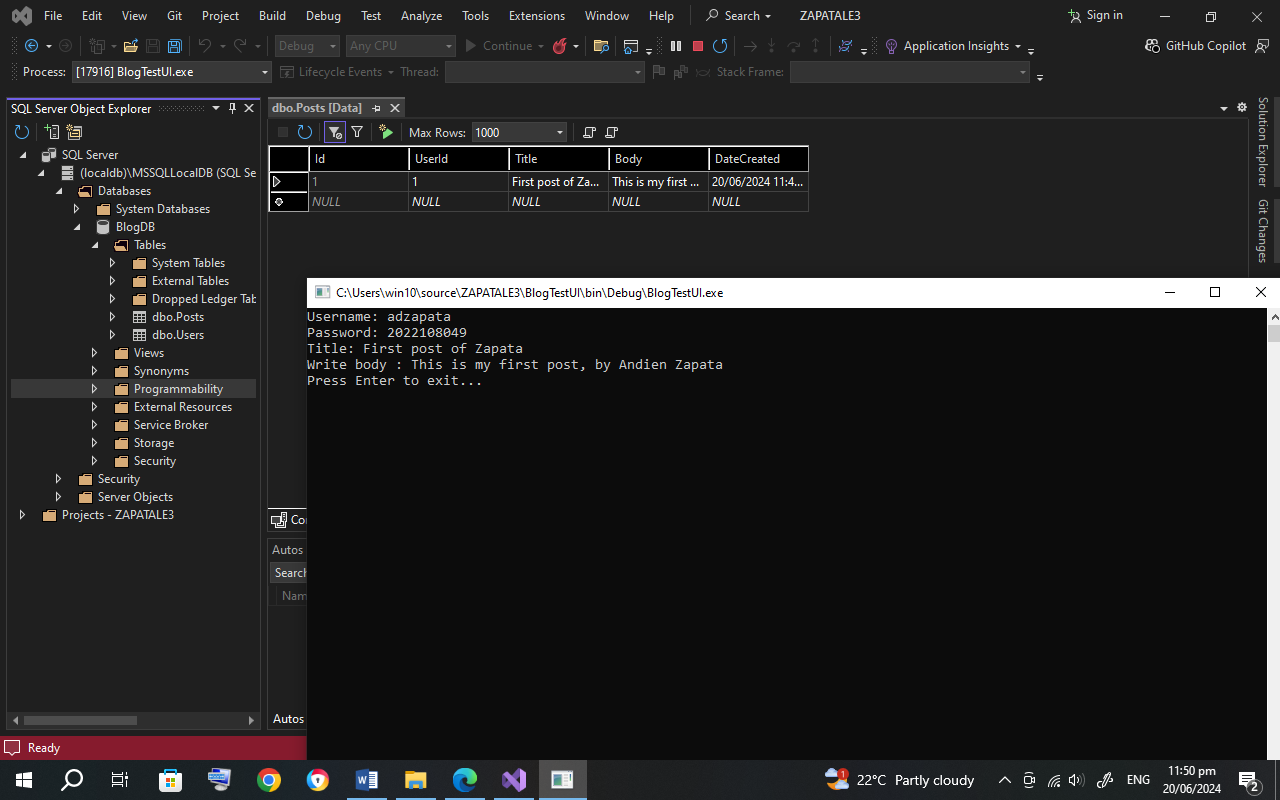




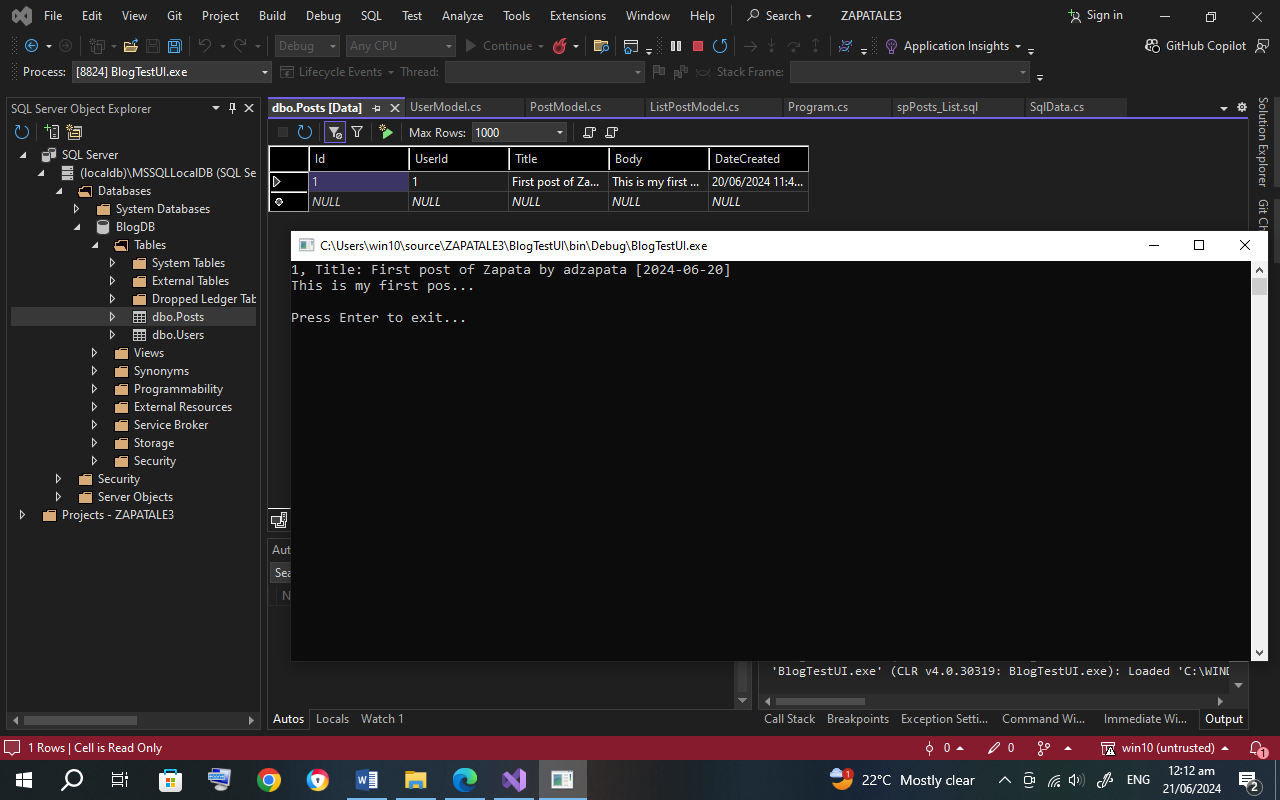
# App functionality: Register



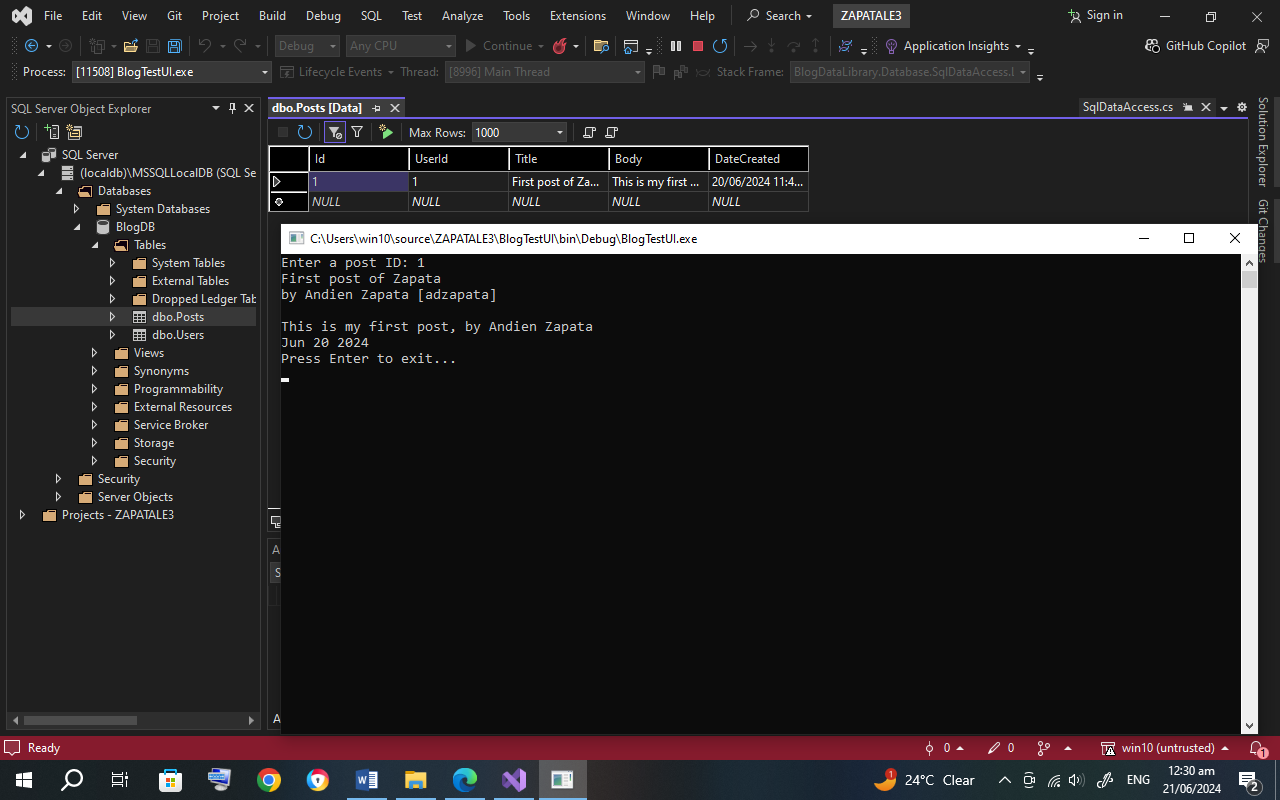
# App functionality: Adding posts



# App functionality: List of posts

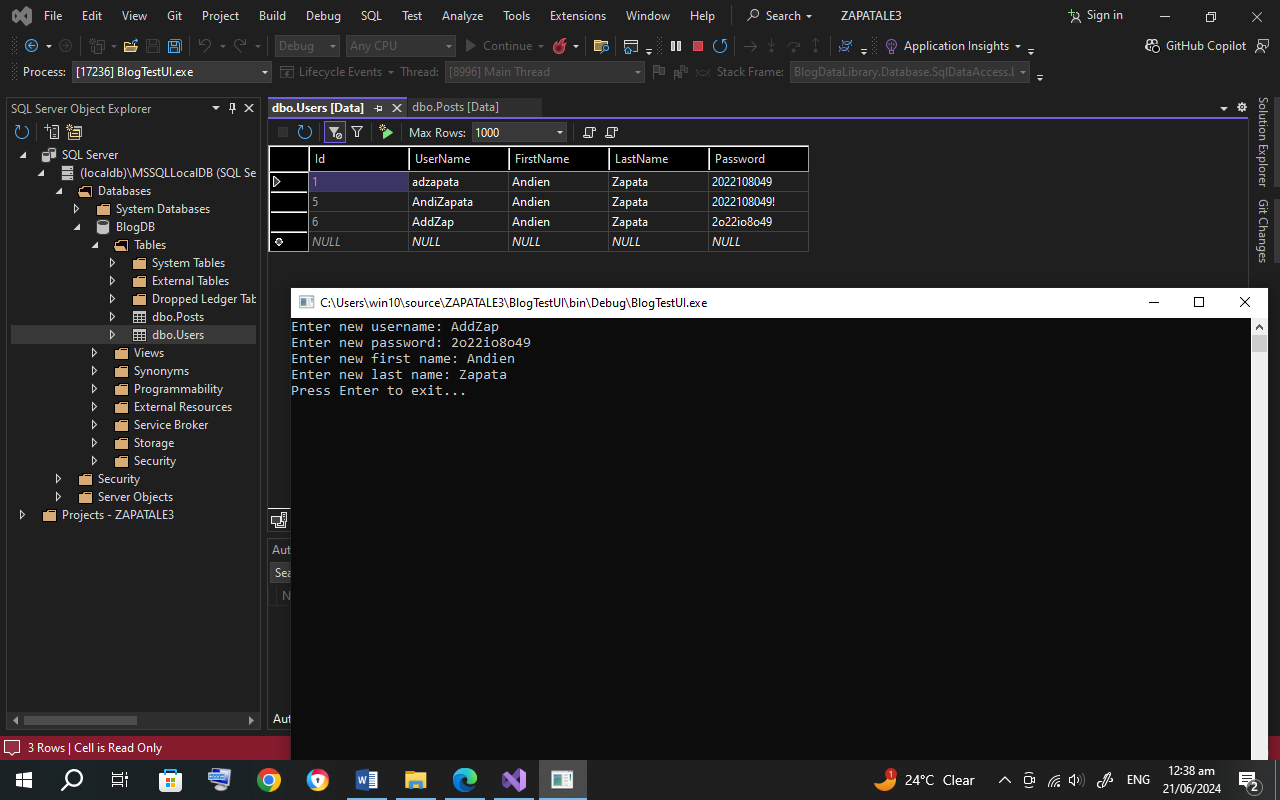


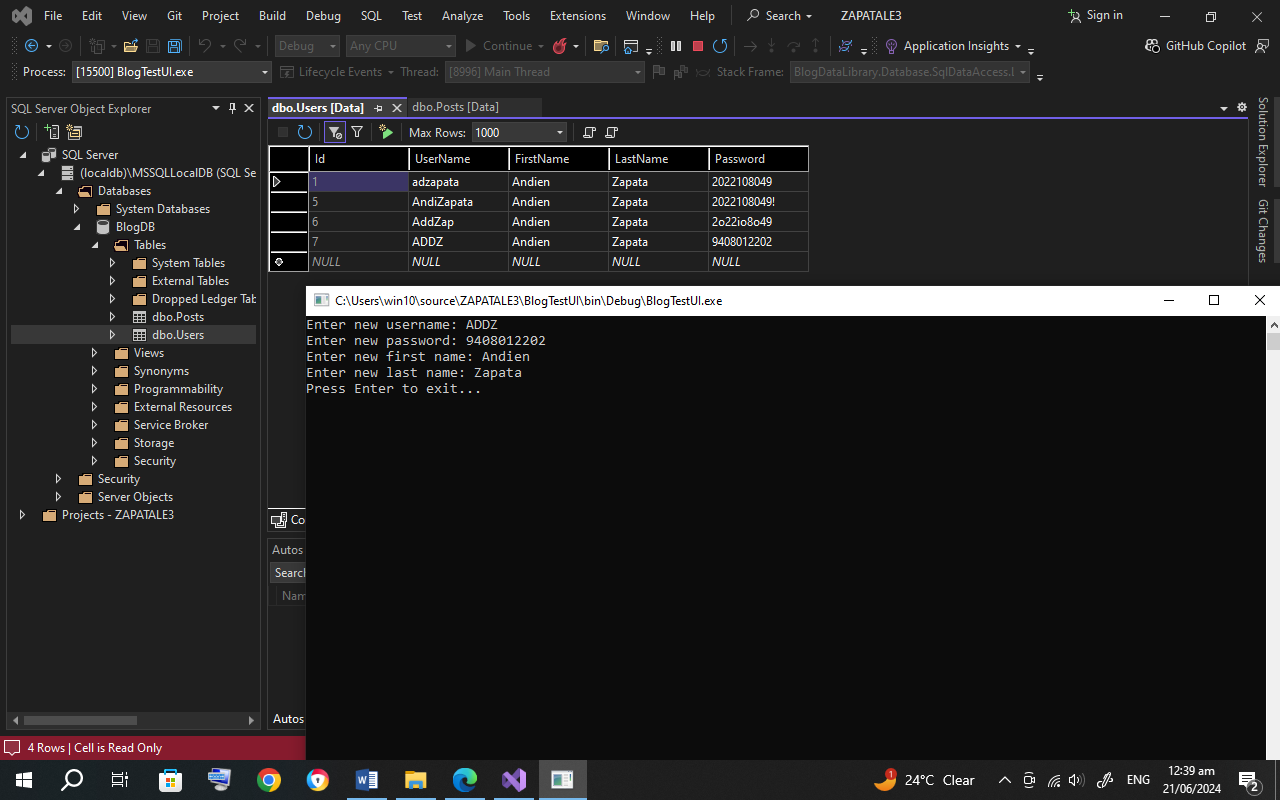
# App functionality: Showing post details

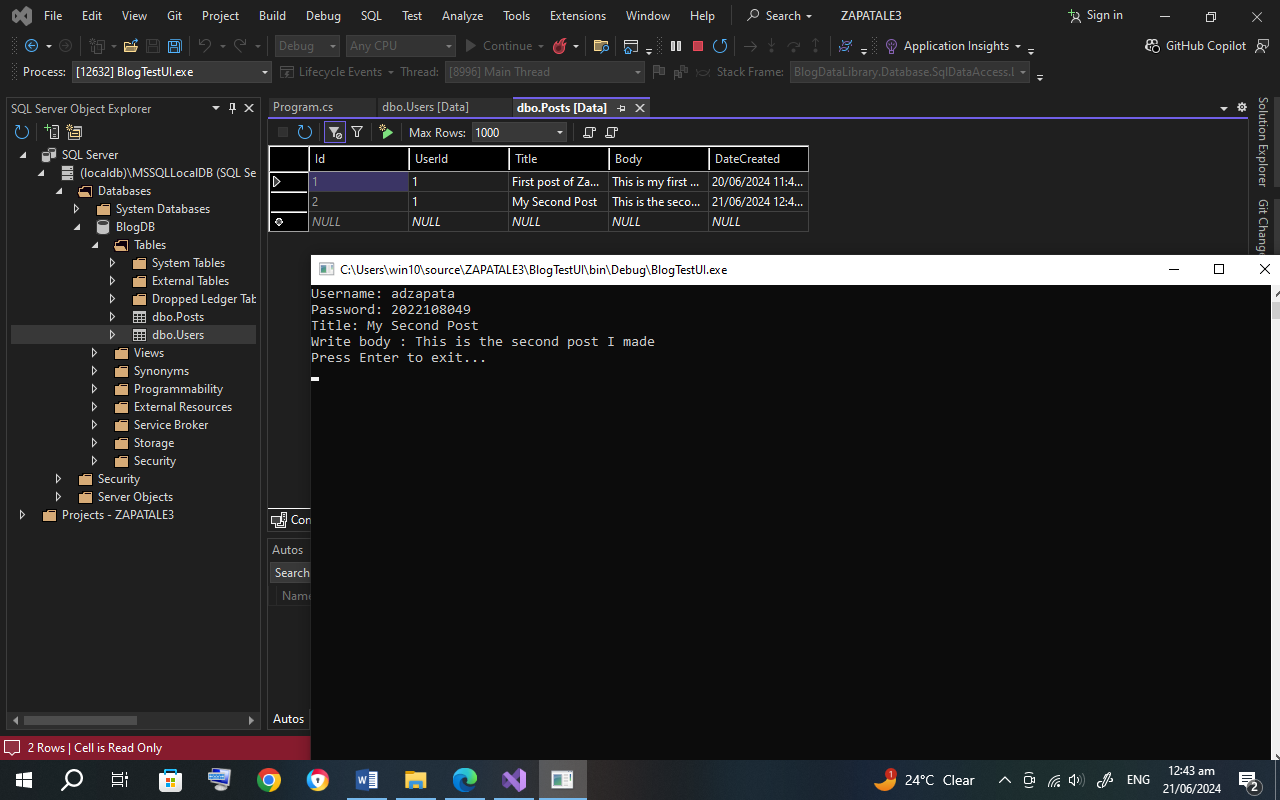


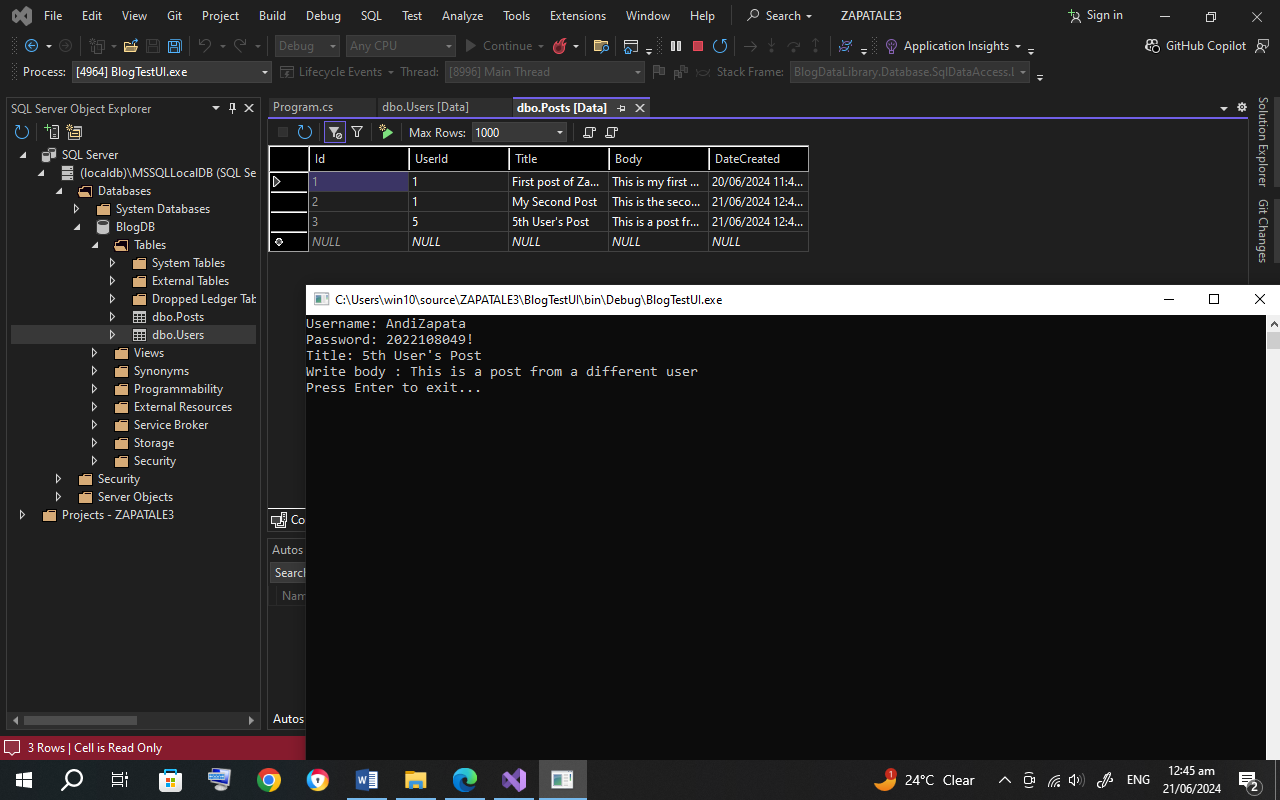
# Post-program requirements

* Add 2 more users and 2 more post. Screenshot the processes and the tables containing the data.

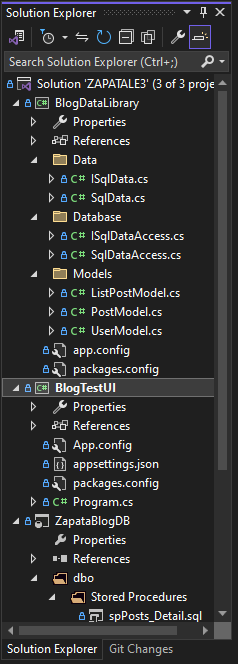
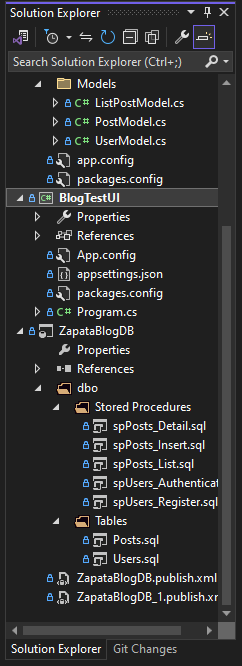








* Screenshot the Solution Explorer. Ensure that all folders are expanded

* Copy and paste all the code in BlogDataLibrary/SqlData.cs and BlogTestUI/Program.cs
* SqlData.cs

using BlogDataLibrary.Database;

using BlogDataLibrary.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BlogDataLibrary.Data

{

public class SqlData : ISqlData

{

private ISqlDataAccess \_db;

private const string connectionStringName = "Sqldb";

public SqlData(ISqlDataAccess db)

{

\_db = db;

}

public UserModel Authenticate(string username, string password)

{

UserModel result = \_db.LoadData<UserModel, dynamic>("dbo.spUsers\_Authenticate", new { username, password }, connectionStringName, true).FirstOrDefault();

return result;

}

public void Register(string username, string firstName, string lastName, string password)

{

\_db.SaveData<dynamic>("dbo.spUsers\_Register", new { username, firstName, lastName, password }, connectionStringName, true);

}

public void AddPost(PostModel post)

{

\_db.SaveData("dbo.spPosts\_Insert", new { post.UserId, post.Title, post.Body, post.DateCreated }, connectionStringName, true);

}

public List<ListPostModel> ListPosts()

{

return \_db.LoadData<ListPostModel, dynamic>("dbo.spPosts\_List", new { }, connectionStringName, true).ToList();

}

public ListPostModel ShowPostDetails(int id)

{

return \_db.LoadData<ListPostModel, dynamic>("dbo.spPosts\_Detail", new { id }, connectionStringName, true).FirstOrDefault();

}

}

}

* Program.cs

using Microsoft.Extensions.Configuration;

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using BlogDataLibrary.Data;

using BlogDataLibrary.Database;

using BlogDataLibrary.Models;

using System.Net.Security;

namespace BlogTestUI

{

internal class Program

{

static void Main(string[] args)

{

SqlData db = GetConnection();

//Authenticate(db);

//Register(db);

//AddPost(db);

//ListPosts(db);

//ShowPostDetails(db);

Console.WriteLine("Press Enter to exit...");

Console.ReadLine();

}

static SqlData GetConnection()

{

var builder = new ConfigurationBuilder().SetBasePath(Directory.GetCurrentDirectory()).AddJsonFile("appsettings.json");

IConfiguration config = builder.Build();

ISqlDataAccess dbAccess = new SqlDataAccess(config);

SqlData db = new SqlData(dbAccess);

return db;

}

private static UserModel GetCurrentUser(SqlData db)

{

Console.Write("Username: ");

string username = Console.ReadLine();

Console.Write("Password: ");

string password = Console.ReadLine();

UserModel user = db.Authenticate(username, password);

return user;

}

public static void Authenticate(SqlData db)

{

UserModel user = GetCurrentUser(db);

if (user == null)

{

Console.WriteLine("Invalid credentials.");

}

else

{

Console.WriteLine($"Welcome, {user.UserName}");

}

}

public static void Register(SqlData db)

{

Console.Write("Enter new username: ");

var username = Console.ReadLine();

Console.Write("Enter new password: ");

var password = Console.ReadLine();

Console.Write("Enter new first name: ");

var firstName = Console.ReadLine();

Console.Write("Enter new last name: ");

var lastName = Console.ReadLine();

db.Register(username, firstName, lastName, password);

}

private static void AddPost(SqlData db)

{

UserModel user = GetCurrentUser(db);

Console.Write("Title: ");

string title = Console.ReadLine();

Console.Write("Write body : ");

string body = Console.ReadLine();

PostModel post = new PostModel

{

Title = title,

Body = body,

DateCreated = DateTime.Now,

UserId = user.Id

};

db.AddPost(post);

}

private static void ListPosts(SqlData db)

{

List<ListPostModel> posts = db.ListPosts();

foreach (ListPostModel post in posts)

{

Console.WriteLine($"{post.Id}, Title: {post.Title} by {post.UserName} [{post.DateCreated.ToString("yyyy-MM-dd")}]");

Console.WriteLine($"{post.Body.Substring(0, 20)}...");

Console.WriteLine();

}

}

private static void ShowPostDetails(SqlData db)

{

Console.Write("Enter a post ID: ");

int id = Int32.Parse(Console.ReadLine());

ListPostModel post = db.ShowPostDetails(id);

Console.WriteLine(post.Title);

Console.WriteLine($"by {post.FirstName} {post.LastName} [{post.UserName}]");

Console.WriteLine();

Console.WriteLine(post.Body);

Console.WriteLine(post.DateCreated.ToString("MMM d yyyy"));

}

}

}

# GitHub link:

<https://github.com/AndiZapata/Zapata-ITS152L>