

**TUGAS JURNAL
MODUL 15**



Disusun Oleh :

Izzaty Zahara Br Barus – 23111040452

Kelas :

SE-07-02

Dosen :

Yudha Islami Sulistya

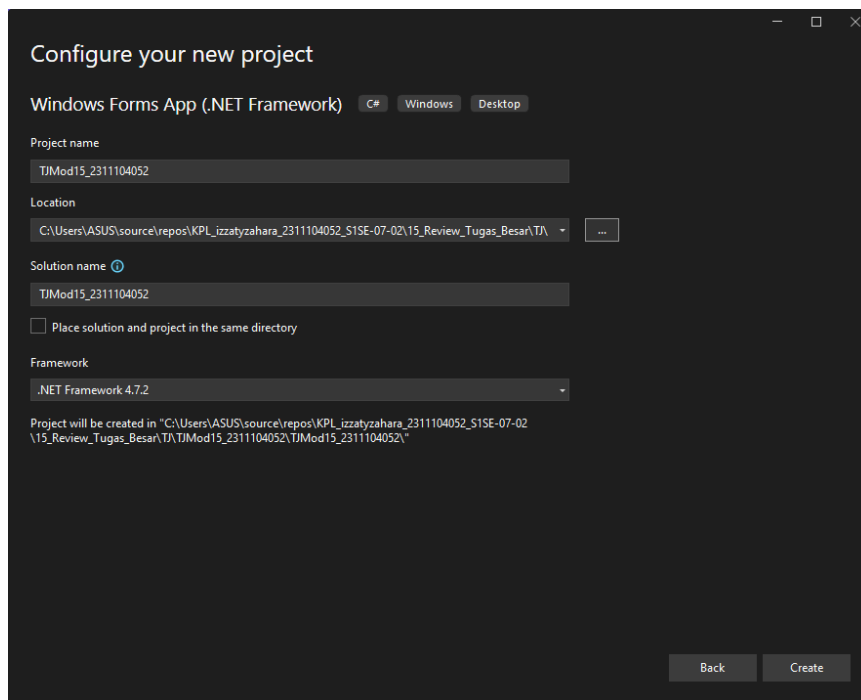
**PROGRAM STUDI SOFTWARE ENGINEERING
DIREKTORAT KAMPUS PURWOKERTO
TELKOM UNIVERSITY PURWOKERTO
2025**

I. Link Github

• https://github.com/Izzaaaaaaaaaa/KPL_izzatyzahara_2311104052_S1SE-07-02.git

II. Alur Pengerjaan

1. Membuat Project Windows froms App TjMod15_2311104052



2. Memasukkan Code pada Class “Formlogin.cs & Formlogin.designer”

```
using System;  
using System.Collections.Generic;  
using System.IO;  
using System.Linq;  
using System.Windows.Forms;
```

```
using Newtonsoft.Json;
using tjmodul15_2311104052.Helpers;
using tjmodul15_2311104052.Models;

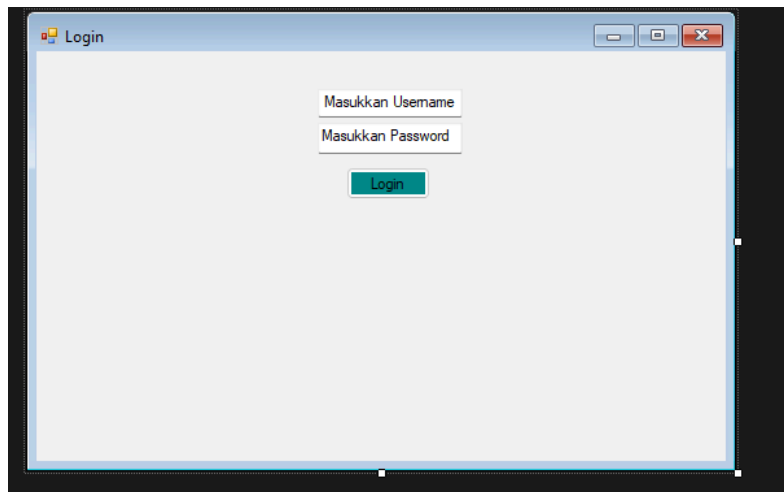
namespace tjmodul15_2311104052
{
    public partial class FormLogin: Form
    {
        private string userFile = "users.json";
        public FormLogin()
        {
            InitializeComponent();
        }

        private void btnLogin_Click(object sender, EventArgs e)
        {
            string username = txtUsername.Text.Trim();
            string inputPassword = txtPassword.Text;

            if (string.IsNullOrEmpty(username) ||
string.IsNullOrEmpty(inputPassword))
            {
                MessageBox.Show("Username dan Password wajib diisi.",
"Peringatan", MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
        }
    }
}
```

```
        string hashedPassword =  
SecurityHelper.HashPassword(inputPassword);  
  
        if (!File.Exists(userFile))  
        {  
            MessageBox.Show("Belum ada user terdaftar.", "Informasi",  
MessageBoxButtons.OK, MessageBoxIcon.Information);  
            return;  
        }  
  
        List<User> users;  
        try  
        {  
            string json = File.ReadAllText(userFile);  
            users = JsonConvert.DeserializeObject<List<User>>(json);  
        }  
        catch (Exception ex)  
        {  
            MessageBox.Show("Gagal membaca file user: " + ex.Message,  
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);  
            return;  
        }  
  
        var user = users.FirstOrDefault(u =>  
            u.Username.Equals(username,
```

```
StringComparison.OrdinalIgnoreCase) &&  
    u.Password == hashedPassword);  
  
    if (user != null)  
    {  
        MessageBox.Show("Login berhasil!", "Sukses",  
        MessageBoxButtons.OK, MessageBoxIcon.Information);  
    }  
    else  
    {  
        MessageBox.Show("Login gagal! Username atau password  
        salah.", "Gagal", MessageBoxButtons.OK, MessageBoxIcon.Error);  
    }  
}  
}
```



Penjelasan Singkat:

Bagian pertama adalah FormLogin, yang berfungsi untuk melakukan proses autentikasi. Pada form ini, user diminta untuk memasukkan username dan password. Validasi dilakukan untuk memastikan kedua input tidak kosong. Password kemudian di-hash menggunakan SHA256 dan dibandingkan dengan data hash yang disimpan dalam users.json. Jika kombinasi username dan hash password cocok, maka login dinyatakan berhasil. Jika tidak cocok, maka muncul pesan bahwa login gagal. File users.json dibaca menggunakan deserialisasi JSON, dan error ditangani dengan try-catch untuk mencegah crash program.

3. Memasukkan Code pada Class “FormRegister.cs & FormRegister.designer”

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text.RegularExpressions;
using System.Windows.Forms;
using Newtonsoft.Json;
using tjmodul15_2311104052.Helpers;
using tjmodul15_2311104052.Models;

namespace tjmodul15_2311104052
{
    public partial class FormRegister: Form
    {
        private readonly string userFile =
Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "users.json");
        public FormRegister()
        {
            InitializeComponent();

            private void btnRegister_Click(object sender, EventArgs e)
```

```
{
    string username = txtUsername.Text.Trim();
    string password = txtPassword.Text;

    if (!IsValidUsername(username))
    {
        MessageBox.Show("Username hanya boleh huruf (3-20 karakter).", "Validasi Gagal", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    if (!IsValidPassword(password, username))
    {
        MessageBox.Show("Password harus 8-20 karakter, mengandung angka & simbol, dan tidak mengandung username.", "Validasi Gagal", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    string hashedPassword =
    SecurityHelper.HashPassword(password);

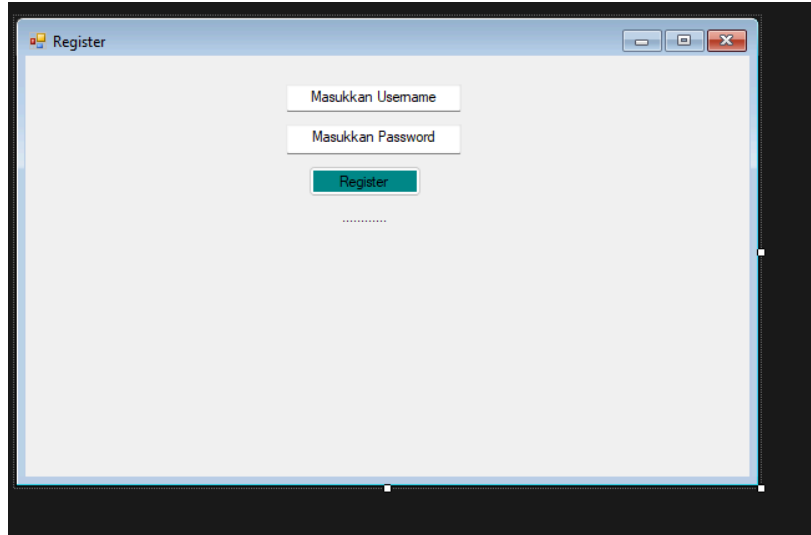
    List<User> users = new List<User>();

    try
    {
        if (File.Exists(userFile))
        {
            string json = File.ReadAllText(userFile);
            if (json.Trim().StartsWith("["))
            {
                users = JsonConvert.DeserializeObject<List<User>>(json);
            }
            else
            {
                MessageBox.Show("Format users.json salah. Harus berupa array JSON.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
        }
    }
```

```
    }  
    }  
    catch (Exception ex)  
    {  
        MessageBox.Show("Gagal membaca users.json: " +  
ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);  
        return;  
    }  
  
    if (users.Exists(u => u.Username.Equals(username,  
StringComparison.OrdinalIgnoreCase)))  
    {  
        MessageBox.Show("Username sudah terdaftar.", "Registrasi  
Gagal", MessageBoxButtons.OK, MessageBoxIcon.Error);  
        return;  
    }  
  
    users.Add(new User { Username = username, Password =  
hashedPassword });  
    File.WriteAllText(userFile, JsonConvert.SerializeObject(users,  
Formatting.Indented));  
  
    MessageBox.Show("Registrasi berhasil!", "Sukses",  
MessageBoxButtons.OK, MessageBoxIcon.Information);  
    this.Close(); // Menutup form setelah registrasi sukses (opsional)  
}  
  
private bool IsValidUsername(string username)  
{  
    return Regex.IsMatch(username, @"^[a-zA-Z]{3,20}$");  
}  
  
private bool IsValidPassword(string password, string username)  
{  
    return password.Length >= 8 &&  
        password.Length <= 20 &&  
        Regex.IsMatch(password, @"[0-9]") &&  
        Regex.IsMatch(password, @"[!@#$%^&*]") &&  
        !password.ToLower().Contains(username.ToLower());  
}
```



```
}  
}
```



Penjelasan singkat:

Bagian kedua adalah FormRegister, yang berfungsi sebagai tempat pendaftaran akun baru. Proses registrasi mencakup validasi username (harus berupa huruf, 3–20 karakter) dan password (8–20 karakter, wajib mengandung angka, simbol, serta tidak mengandung bagian dari username). Jika validasi berhasil, password di-hash dan data user ditambahkan ke file users.json. Sebelum data disimpan, sistem memastikan bahwa username belum pernah terdaftar. Seluruh proses disertai dengan pesan peringatan atau sukses sesuai hasilnya. Form ini juga menggunakan helper tambahan yaitu SecurityHelper untuk hashing password, serta model User yang memuat properti Username dan Password.

4. Memasukkan Code pada Class “SecurityHelper.cs”

```
using System.Security.Cryptography;
using System.Text;

namespace tjmodul15_2311104052.Helpers
{
    public static class SecurityHelper
    {
        public static string HashPassword(string password)
        {
            using (SHA256 sha = SHA256.Create())
            {
                byte[] bytes =
                sha.ComputeHash(Encoding.UTF8.GetBytes(password));
                StringBuilder builder = new StringBuilder();
                foreach (byte b in bytes)
                    builder.Append(b.ToString("x2"));
                return builder.ToString();
            }
        }
    }
}
```

Penjelasan singkat:

Bagian ketiga adalah implementasi SecurityHelper.cs yang menyediakan fungsi hashing password dengan algoritma SHA256. Fungsi ini digunakan baik oleh FormLogin maupun FormRegister untuk memastikan bahwa data password yang tersimpan dan dibandingkan selalu dalam bentuk hash. Selanjutnya, terdapat class model User di dalam folder Models, yang berfungsi untuk merepresentasikan data user dengan dua properti: Username dan Password.

5. Memasukkan Code pada Class “Models.cs”

```
namespace tjmodul15_2311104052.Models
```

```
{  
  public class User  
  {  
    public string Username { get; set; }  
    public string Password { get; set; }  
  }  
}
```

Penjelasan singkat:

Sementara itu, file users.json digunakan untuk menyimpan seluruh data user dalam bentuk array JSON. Contoh isinya berupa daftar objek dengan pasangan Username dan Password (dalam bentuk hash). File ini dibaca dan ditulis secara otomatis saat proses login dan registrasi berlangsung.

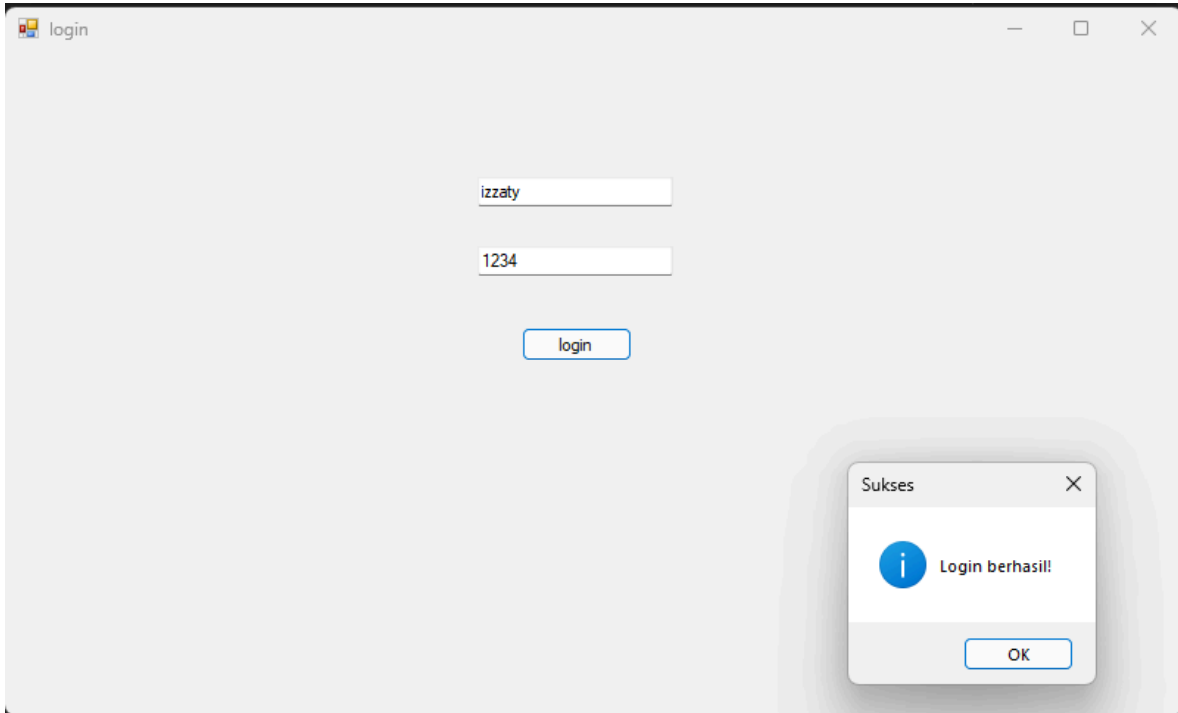
6. Memasukkan Code pada Class “User .json”

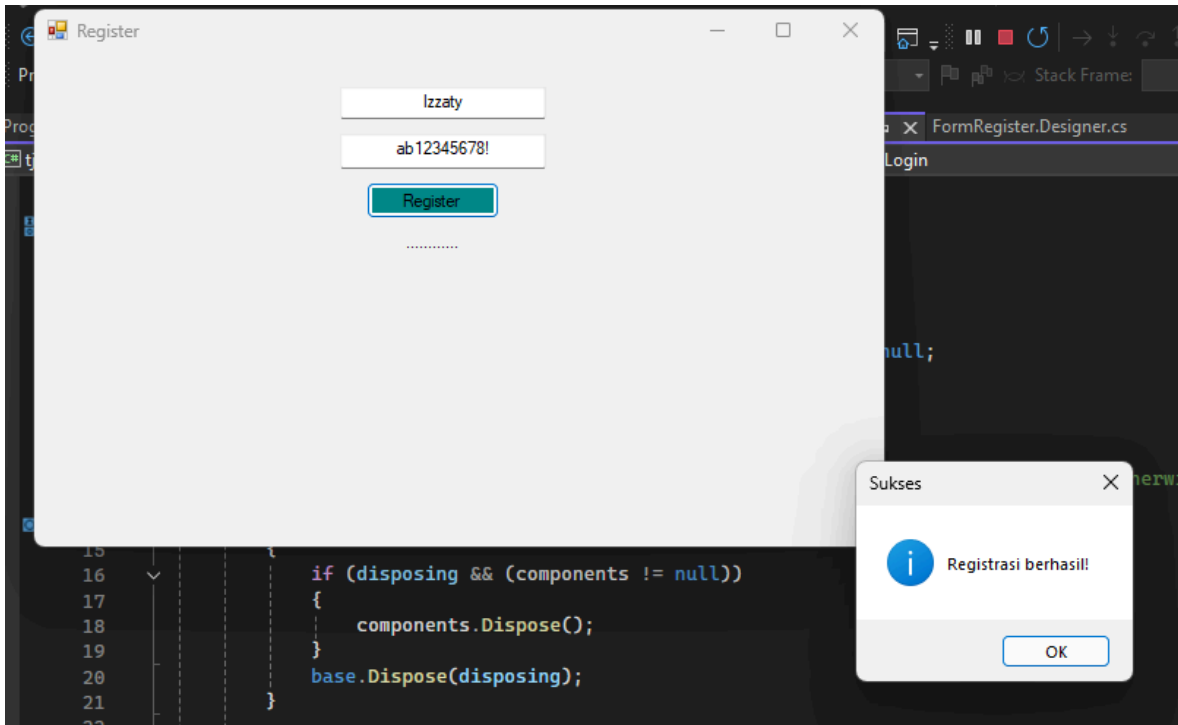
```
[  
  {  
    "Username": "admin",  
    "Password": "e99a18c428cb38f708663b87029e4abed8134aa"  
  }  
]
```

Penjelasan singkat:

Pada bagian akhir, ditampilkan output keberhasilan login menggunakan akun izzaty dengan password 1234, dan hasil registrasi akun baru. Semua fungsionalitas diuji melalui antarmuka form yang telah didesain menggunakan komponen TextBox, Button, dan Label.

III. Hasil Output





IV. Kesimpulan

Melalui pengerjaan Tugas Jurnal Modul 15 ini, dapat disimpulkan bahwa penerapan prinsip secure coding sangat penting dalam membangun sistem autentikasi, bahkan untuk aplikasi desktop yang sederhana seperti Windows Forms App. Proyek ini mengintegrasikan dua fitur utama yaitu login dan register, yang masing-masing dilengkapi dengan validasi input dan perlindungan data menggunakan teknik hashing SHA256. Hal ini menunjukkan pemahaman praktis mengenai bagaimana menjaga kerahasiaan data pengguna, khususnya password, agar tidak disimpan dalam bentuk asli (plaintext), melainkan dalam bentuk hash yang tidak mudah dibalikkan.

Selain itu, penggunaan file JSON sebagai media penyimpanan data user menunjukkan bahwa pengelolaan data tidak selalu membutuhkan database, tetapi tetap bisa dilakukan secara efisien dan terstruktur asalkan formatnya benar dan aman.

Implementasi class `SecurityHelper` dan `User` sebagai modul pendukung juga memperlihatkan kemampuan dalam membuat kode yang terorganisir dan reusable, yang merupakan bagian penting dari praktik pemrograman modern.

Dalam prosesnya, validasi yang ketat terhadap username dan password berhasil mencegah input yang tidak sesuai, seperti username yang terlalu pendek atau password yang tidak aman. Ini sejalan dengan prinsip dasar pengamanan aplikasi dari sisi input user. Keseluruhan proses dari registrasi hingga login mampu berjalan lancar dan memberikan umpan balik (feedback) yang jelas kepada pengguna melalui notifikasi yang informatif.

Dengan menyelesaikan tugas ini, penulis mendapatkan pengalaman langsung dalam mengembangkan sistem autentikasi yang aman dan memahami pentingnya konsistensi dalam struktur data (misalnya format JSON), penanganan error, serta penerapan good practice dalam membangun aplikasi berbasis GUI. Sistem ini dapat menjadi pondasi awal untuk pengembangan aplikasi berskala lebih besar dan kompleks di masa mendatang, dengan tambahan fitur seperti pengaturan peran user, koneksi ke database, atau otentikasi multi-level. Dengan demikian, tugas ini tidak hanya melatih kemampuan teknis, tetapi juga memperkuat kesadaran akan pentingnya keamanan dalam pengembangan perangkat lunak.