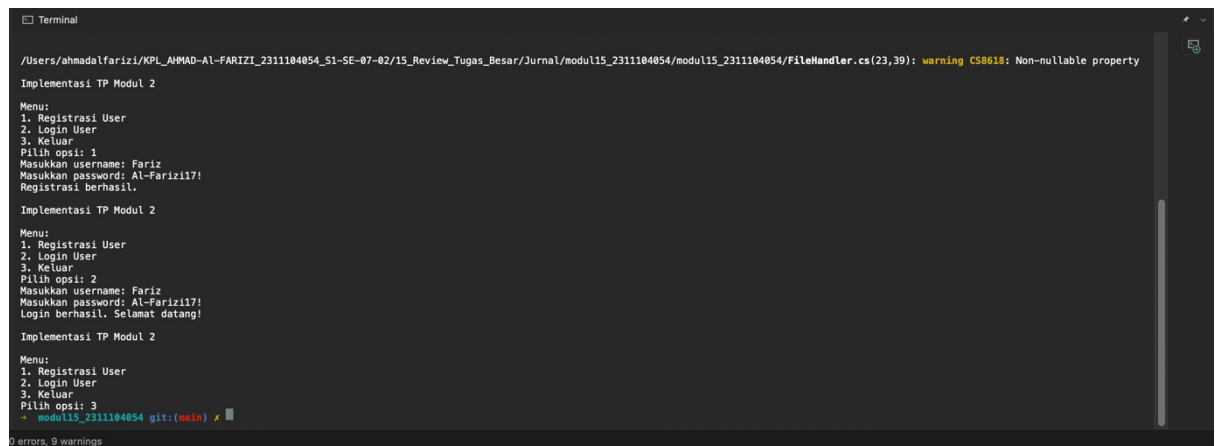


Nama: Ahmad Al – Farizi

NIM: 2311104054

- i. Link GitHub: https://github.com/alxfarizi/KPL_AHMAD-AL-FARIZI_2311104054_S1SE-07-02/tree/main/15_Review_Tugas_Besar/Jurnal/modul15_2311104054/modul15_2311104054

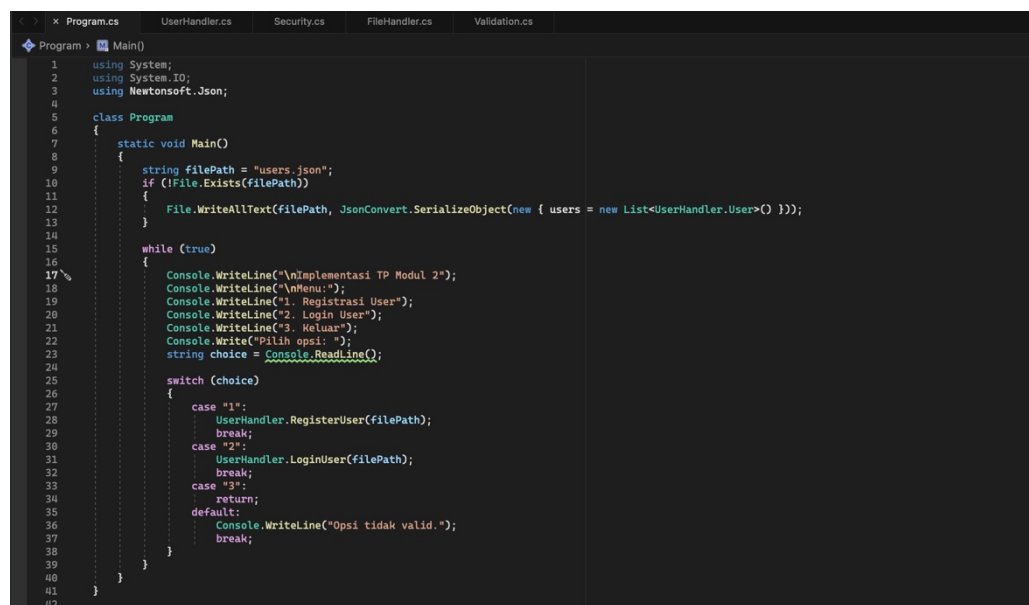
- ii. Hasil Running:



```
Terminal
/Users/ahmadalfarizi/KPL_AHMAD-AL-FARIZI_2311104054_S1-SE-07-02/15_Review_Tugas_Besar/Jurnal/modul15_2311104054/modul15_2311104054/FileHandler.cs(23,39): warning CS8618: Non-nullable property
Implementasi TP Modul 2
Menu:
1. Registrasi User
2. Login User
3. Keluar
Pilih opsi: 1
Masukkan username: Fariz
Masukkan password: AL-Farizi17!
Registrasi berhasil.
Implementasi TP Modul 2
Menu:
1. Registrasi User
2. Login User
3. Keluar
Pilih opsi: 2
Masukkan username: Fariz
Masukkan password: AL-Farizi17!
Login berhasil. Selamat datang!
Implementasi TP Modul 2
Menu:
1. Registrasi User
2. Login User
3. Keluar
Pilih opsi: 3
modul15_2311104054 git:(main) x
```

- iii. Penjelasan Source Code:

- a. Program.cs, File utama ini berfungsi sebagai titik awal aplikasi. Program.cs menampilkan menu utama kepada pengguna untuk memilih antara fitur registrasi, login, atau keluar dari program. Fungsi registrasi dan login diimplementasikan dengan memanggil metode dari kelas UserHandler.



```
Program.cs
1 using System;
2 using System.IO;
3 using Newtonsoft.Json;
4
5 class Program
6 {
7     static void Main()
8     {
9         string filePath = "users.json";
10        if (!File.Exists(filePath))
11        {
12            File.WriteAllText(filePath, JsonConvert.SerializeObject(new { users = new List<UserHandler.User>() }));
13        }
14
15        while (true)
16        {
17            Console.WriteLine("\nImplementasi TP Modul 2");
18            Console.WriteLine("\nMenu:");
19            Console.WriteLine("1. Registrasi User");
20            Console.WriteLine("2. Login User");
21            Console.WriteLine("3. Keluar");
22            Console.WriteLine("Pilih opsi: ");
23            string choice = Console.ReadLine();
24
25            switch (choice)
26            {
27                case "1":
28                    UserHandler.RegisterUser(filePath);
29                    break;
30                case "2":
31                    UserHandler.LoginUser(filePath);
32                    break;
33                case "3":
34                    return;
35                default:
36                    Console.WriteLine("Ops! tidak valid.");
37                    break;
38            }
39        }
40    }
41 }
42
```

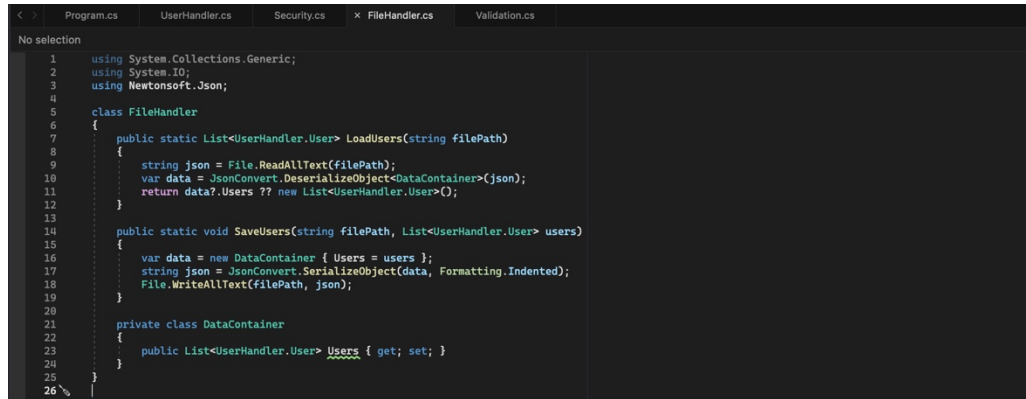
- b. UserHandler.cs, File ini mengatur logika utama untuk fitur registrasi dan login. Pada registrasi, username dan password divalidasi menggunakan metode dari kelas Validation, dan password di-hash menggunakan kelas Security sebelum disimpan ke file JSON melalui FileHandler. Untuk login, file ini membandingkan username dan password yang dimasukkan dengan data yang tersimpan.

```
< > Program.cs x UserHandler.cs Security.cs FileHandler.cs Validation.cs
No selection
1 using System;
2 using System.Collections.Generic;
3 using System.IO;
4 using System.Security.Cryptography;
5 using System.Text;
6 using Newtonsoft.Json;
7
8 class UserHandler
9 {
10     public static void RegisterUser(string filePath)
11     {
12         Console.Write("Masukkan username: ");
13         string username = Console.ReadLine();
14
15         if (!Validation.IsValidUsername(username))
16         {
17             Console.WriteLine("Username hanya boleh huruf alfabet ASCII.");
18             return;
19         }
20
21         Console.Write("Masukkan password: ");
22         string password = Console.ReadLine();
23
24         if (!Validation.IsValidPassword(password))
25         {
26             Console.WriteLine("Password harus 8-20 karakter, mengandung angka dan karakter unik (!@#$%^&*~).");
27             return;
28         }
29
30         string hashedPassword = Security.HashPassword(password);
31
32         var users = FileHandler.LoadUsers(filePath);
33         users.Add(new User { Username = username, Password = hashedPassword });
34         FileHandler.SaveUsers(filePath, users);
35
36         Console.WriteLine("Registrasi berhasil.");
37     }
38
39     public static void LoginUser(string filePath)
40     {
41         Console.Write("Masukkan username: ");
42         string username = Console.ReadLine();
43
44         Console.Write("Masukkan password: ");
45         string password = Console.ReadLine();
46
47         string hashedPassword = Security.HashPassword(password);
48
49         var users = FileHandler.LoadUsers(filePath);
50         foreach (var user in users)
51         {
52             if (user.Username == username && user.Password == hashedPassword)
53             {
54                 Console.WriteLine("Login berhasil. Selamat datang!");
55                 return;
56             }
57         }
58
59         Console.WriteLine("Login gagal. Username atau password salah.");
60     }
61
62     public class User
63     {
64         public string? Username { get; set; }
65         public string? Password { get; set; }
66     }
67 }
```

- c. Validation.cs, File ini bertanggung jawab memvalidasi data masukan seperti username dan password. Validasi mencakup pemeriksaan apakah username hanya berisi huruf alfabet dan apakah password memenuhi aturan panjang (8-20 karakter), mengandung angka, serta karakter unik.

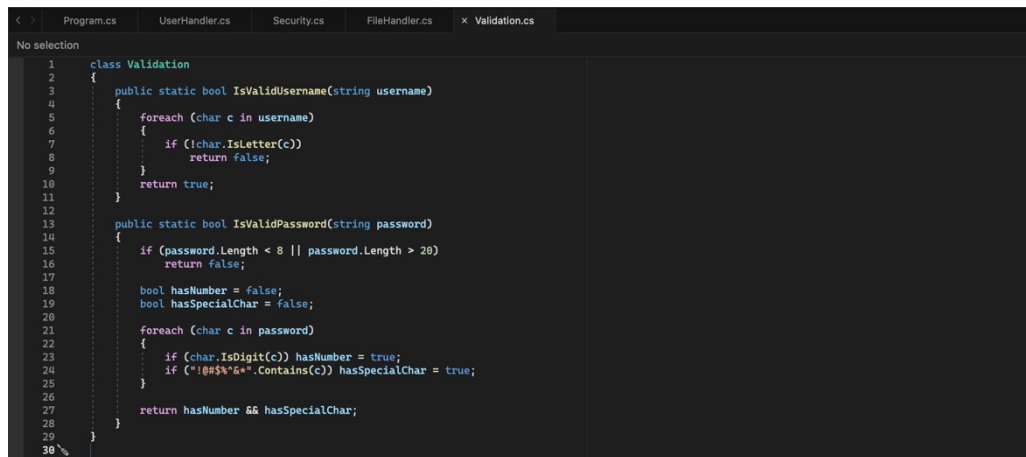
```
< > Program.cs UserHandler.cs x Security.cs FileHandler.cs Validation.cs
No selection
1 using System.Security.Cryptography;
2 using System.Text;
3
4 class Security
5 {
6     public static string HashPassword(string password)
7     {
8         using (SHA256 sha256 = SHA256.Create())
9         {
10             byte[] bytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));
11             StringBuilder builder = new StringBuilder();
12             foreach (byte b in bytes)
13             {
14                 builder.Append(b.ToString("x2"));
15             }
16             return builder.ToString();
17         }
18     }
19 }
20 }
```

- d. Security.cs, File ini menangani hashing password menggunakan algoritma SHA256. Hashing memastikan bahwa password yang disimpan di database atau file JSON tidak berupa teks asli, sehingga lebih aman jika data disusupi.



```
1 using System.Collections.Generic;
2 using System.IO;
3 using Newtonsoft.Json;
4
5 class FileHandler
6 {
7     public static List<UserHandler.User> LoadUsers(string filePath)
8     {
9         string json = File.ReadAllText(filePath);
10        var data = JsonConvert.DeserializeObject<DataContainer>(json);
11        return data?.Users ?? new List<UserHandler.User>();
12    }
13
14    public static void SaveUsers(string filePath, List<UserHandler.User> users)
15    {
16        var data = new DataContainer { Users = users };
17        string json = JsonConvert.SerializeObject(data, Formatting.Indented);
18        File.WriteAllText(filePath, json);
19    }
20
21    private class DataContainer
22    {
23        public List<UserHandler.User> Users { get; set; }
24    }
25 }
26 \
```

- e. FileHandler.cs, File ini mengelola pembacaan dan penulisan data user ke file JSON. Data user diformat dan disimpan dengan struktur JSON, serta diload kembali saat diperlukan. File ini menggunakan pustaka Newtonsoft.Json untuk mempermudah manipulasi data JSON.



```
1 class Validation
2 {
3     public static bool IsValidUsername(string username)
4     {
5         foreach (char c in username)
6         {
7             if (!char.IsLetter(c))
8                 return false;
9         }
10        return true;
11    }
12
13    public static bool IsValidPassword(string password)
14    {
15        if (password.Length < 8 || password.Length > 20)
16            return false;
17
18        bool hasNumber = false;
19        bool hasSpecialChar = false;
20
21        foreach (char c in password)
22        {
23            if (char.IsDigit(c)) hasNumber = true;
24            if ("!@#$%^&*".Contains(c)) hasSpecialChar = true;
25        }
26
27        return hasNumber && hasSpecialChar;
28    }
29 }
30 \
```