

Nama: Marvel Sanjaya Setiawan

NIM: 2311104053

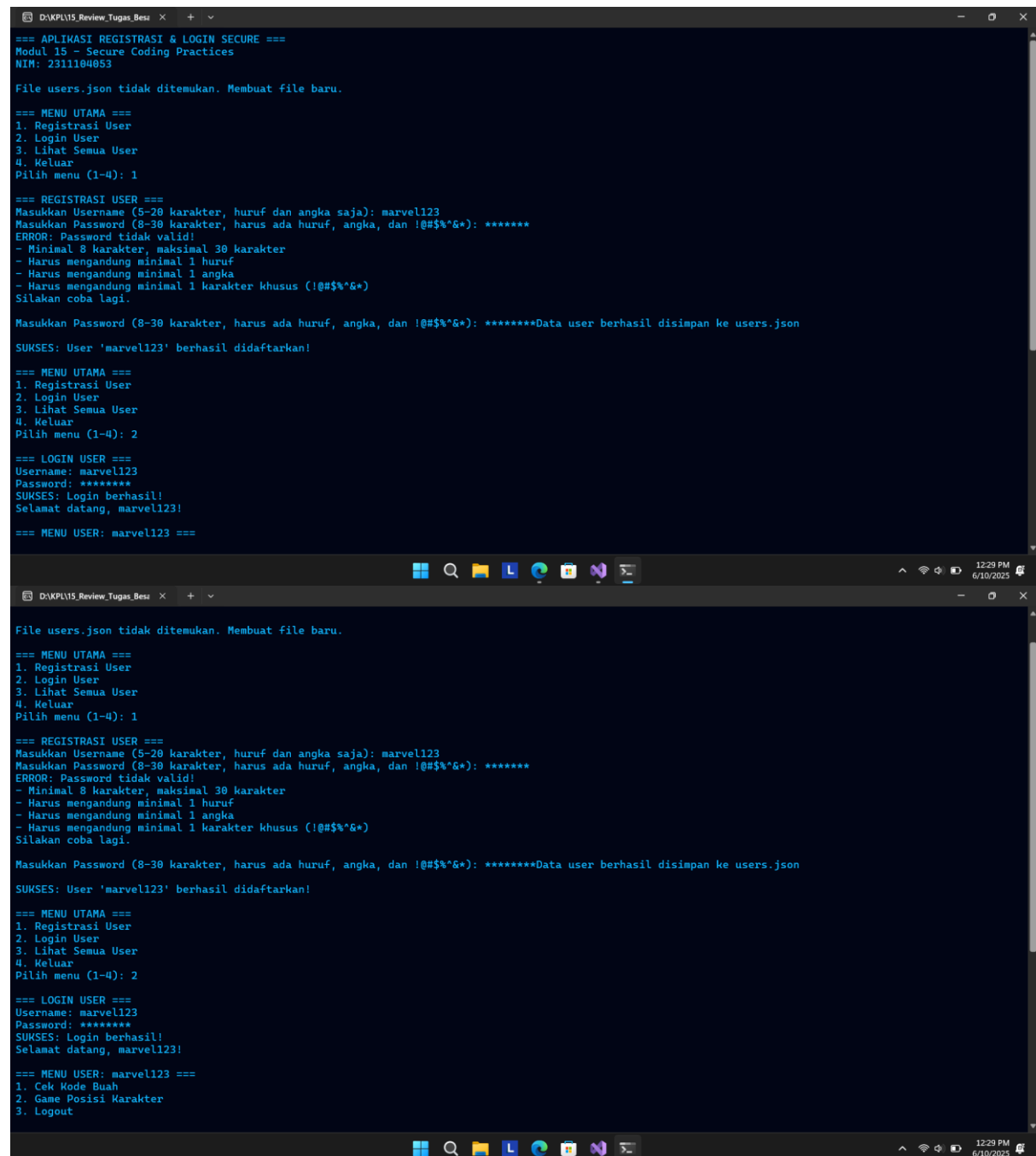
Kelas: SE07-02

## JURNAL MODUL 15

Link github:

[https://github.com/Meph1sto14/KPL\\_Marvel\\_Sanjaya\\_Setiawan\\_2311104053\\_SE07-02/tree/e93333782673e31ffd0185421cd92507c5e4b863/15\\_Review\\_Tugas\\_Besar/JURNAL/modul15\\_2311104053](https://github.com/Meph1sto14/KPL_Marvel_Sanjaya_Setiawan_2311104053_SE07-02/tree/e93333782673e31ffd0185421cd92507c5e4b863/15_Review_Tugas_Besar/JURNAL/modul15_2311104053)

Hasil run:



```
D:\KPL15_Review_Tugas_Besr x + v
=== APLIKASI REGISTRASI & LOGIN SECURE ===
Modul 15 - Secure Coding Practices
NIM: 2311104053

File users.json tidak ditemukan. Membuat file baru.

=== MENU UTAMA ===
1. Registrasi User
2. Login User
3. Lihat Semua User
4. Keluar
Pilih menu (1-4): 1

=== REGISTRASI USER ===
Masukkan Username (5-20 karakter, huruf dan angka saja): marvel123
Masukkan Password (8-30 karakter, harus ada huruf, angka, dan !@#%*&): *****
ERROR: Password tidak valid!
- Minimal 8 karakter, maksimal 30 karakter
- Harus mengandung minimal 1 huruf
- Harus mengandung minimal 1 angka
- Harus mengandung minimal 1 karakter khusus (!@#%*&)
Silakan coba lagi.

Masukkan Password (8-30 karakter, harus ada huruf, angka, dan !@#%*&): *****Data user berhasil disimpan ke users.json
SUKSES: User 'marvel123' berhasil didaftarkan!

=== MENU UTAMA ===
1. Registrasi User
2. Login User
3. Lihat Semua User
4. Keluar
Pilih menu (1-4): 2

=== LOGIN USER ===
Username: marvel123
Password: *****
SUKSES: Login berhasil!
Selamat datang, marvel123!

=== MENU USER: marvel123 ===
1. Cek Kode Buah
2. Game Posisi Karakter
3. Logout
```

## KodeBuah.cs

```
1. using System;
2. using System.Collections.Generic;
3.
4. namespace modul15_2311104053
5. {
6.     public class KodeBuah
7.     {
8.         private static Dictionary<string, string> tabelKodeBuah = new Dictionary<string,
string>
9.         {
10.             {"Apel", "A00"},
11.             {"Aprikot", "B00"},
12.             {"Alpukat", "C00"},
13.             {"Pisang", "D00"},
14.             {"Paprika", "E00"},
15.             {"Blackberry", "F00"},
16.             {"Ceri", "H00"},
17.             {"Kelapa", "I00"},
18.             {"Jagung", "J00"},
19.             {"Kurma", "K00"},
20.             {"Durian", "L00"},
21.             {"Anggur", "M00"},
22.             {"Melon", "N00"},
23.             {"Semangka", "O00"}
24.         };
25.
26.         public static string GetKodeBuah(string namaBuah)
27.         {
28.             return tabelKodeBuah.ContainsKey(namaBuah) ? tabelKodeBuah[namaBuah] : "Kode
tidak ditemukan";
29.         }
30.     }
31. }
32.
```

## PosisiKarakterGame.cs

```
1. using System;
2.
3. namespace modul15_2311104053
4. {
5.     public class PosisiKarakterGame
6.     {
7.         private string state = "Berdiri";
8.         private readonly int nimMod;
9.
10.        public PosisiKarakterGame(long nim)
11.        {
12.            nimMod = (int)(nim % 3); // Menghitung hasil mod dari NIM
13.        }
14.
15.        public void TekanTombol(char tombol)
16.        {
17.            if (nimMod == 0)
18.            {
19.                if (tombol == 'S') Console.WriteLine("Tombol arah bawah ditekan");
20.                if (tombol == 'W') Console.WriteLine("Tombol arah atas ditekan");
21.            }
22.
23.            if (state == "Berdiri" && tombol == 'S')
24.            {
25.                state = "Jongkok";
26.                Console.WriteLine("State berubah menjadi: Jongkok");
27.            }
28.            else if (state == "Jongkok" && tombol == 'W')
```

```

29.         {
30.             state = "Berdiri";
31.             Console.WriteLine("State berubah menjadi: Berdiri");
32.             if (nimMod == 1) Console.WriteLine("Posisi standby");
33.         }
34.         else if (state == "Jongkok" && tombol == 'X')
35.         {
36.             state = "Terbang";
37.             Console.WriteLine("State berubah menjadi: Terbang");
38.             if (nimMod == 2) Console.WriteLine("Posisi take off");
39.         }
40.         else if (state == "Terbang" && tombol == 'S')
41.         {
42.             state = "Berdiri";
43.             Console.WriteLine("State berubah menjadi: Berdiri");
44.         }
45.         else if (state == "Terbang" && tombol == 'J')
46.         {
47.             state = "Jongkok";
48.             Console.WriteLine("State berubah menjadi: Jongkok");
49.             if (nimMod == 2) Console.WriteLine("Posisi landing");
50.         }
51.         else if (state == "Berdiri" && tombol == 'W')
52.         {
53.             state = "Tengkurap";
54.             Console.WriteLine("State berubah menjadi: Tengkurap");
55.             if (nimMod == 1) Console.WriteLine("Posisi istirahat");
56.         }
57.         else
58.         {
59.             Console.WriteLine("Input tombol tidak valid!");
60.         }
61.     }
62.
63.     public void PrintState()
64.     {
65.         Console.WriteLine($"State saat ini: {state}");
66.     }
67. }
68. }
69.

```

## User.cs

```

1. using System;
2. using System.Linq;
3.
4. namespace modul15_2311104053
5. {
6.     public class User
7.     {
8.         public string Username { get; set; }
9.         public string PasswordHash { get; set; }
10.
11.         public User()
12.         {
13.         }
14.
15.         public User(string username, string passwordHash)
16.         {
17.             Username = username;
18.             PasswordHash = passwordHash;
19.         }
20.
21.         // Validasi username
22.         public static bool IsValidUsername(string username)
23.         {

```

```

24.         if (string.IsNullOrEmpty(username))
25.             return false;
26.
27.         // Input Validation - Range dan Panjang data
28.         // Username harus 5-20 karakter, hanya huruf dan angka
29.         if (username.Length < 5 || username.Length > 20)
30.             return false;
31.
32.         // Hanya boleh huruf alfabet ASCII dan angka
33.         foreach (char c in username)
34.         {
35.             if (!char.IsLetterOrDigit(c))
36.                 return false;
37.         }
38.
39.         return true;
40.     }
41.
42.     // Validasi password sesuai Password Rules
43.     public static bool IsValidPassword(string password)
44.     {
45.         if (string.IsNullOrEmpty(password))
46.             return false;
47.
48.         // Password Management - Password rules
49.         // Minimal 8 karakter, maksimal 30 karakter
50.         if (password.Length < 8 || password.Length > 30)
51.             return false;
52.
53.         // Harus mengandung minimal 1 angka
54.         bool hasDigit = false;
55.         // Harus mengandung minimal 1 karakter unik
56.         bool hasSpecialChar = false;
57.         // Harus mengandung minimal 1 huruf
58.         bool hasLetter = false;
59.
60.         string specialChars = "!@#$%^&*";
61.
62.         foreach (char c in password)
63.         {
64.             if (char.IsDigit(c))
65.                 hasDigit = true;
66.             else if (char.IsLetter(c))
67.                 hasLetter = true;
68.             else if (specialChars.Contains(c))
69.                 hasSpecialChar = true;
70.         }
71.
72.         return hasDigit && hasSpecialChar && hasLetter;
73.     }
74.
75.     // Validasi password tidak mengandung username
76.     public static bool PasswordContainsUsername(string password, string username)
77.     {
78.         if (string.IsNullOrEmpty(password) || string.IsNullOrEmpty(username))
79.             return false;
80.
81.         return password.ToLower().Contains(username.ToLower());
82.     }
83. }
84. }
85.

```

## Program.cs

```
1. using System;
2. using System.Collections.Generic;
3. using System.IO;
4. using System.Linq;
5. using System.Security.Cryptography;
6. using System.Text;
7.
8. namespace modul15_2311104053
9. {
10.     class Program
11.     {
12.         private static readonly string UsersFilePath =
13. @"D:\KPL\15_Review_Tugas_Besar\JURNAL\modul15_2311104053\modul15_2311104053\users.json";
14.         private static List<User> users = new List<User>();
15.
16.         static void Main(string[] args)
17.         {
18.             Console.WriteLine("=== APLIKASI REGISTRASI & LOGIN SECURE ===");
19.             Console.WriteLine("Modul 15 - Secure Coding Practices");
20.             Console.WriteLine("NIM: 2311104053\n");
21.
22.             // Load existing users dari file JSON
23.             LoadUsersFromFile();
24.
25.             while (true)
26.             {
27.                 ShowMainMenu();
28.                 string choice = Console.ReadLine();
29.
30.                 switch (choice)
31.                 {
32.                     case "1":
33.                         RegisterUser();
34.                         break;
35.                     case "2":
36.                         LoginUser();
37.                         break;
38.                     case "3":
39.                         ShowAllUsers();
40.                         break;
41.                     case "4":
42.                         Console.WriteLine("Terima kasih! Aplikasi selesai.");
43.                         return;
44.                     default:
45.                         Console.WriteLine("Pilihan tidak valid! Silakan coba lagi.\n");
46.                         break;
47.                 }
48.             }
49.
50.             static void ShowMainMenu()
51.             {
52.                 Console.WriteLine("=== MENU UTAMA ===");
53.                 Console.WriteLine("1. Registrasi User");
54.                 Console.WriteLine("2. Login User");
55.                 Console.WriteLine("3. Lihat Semua User");
56.                 Console.WriteLine("4. Keluar");
57.                 Console.Write("Pilih menu (1-4): ");
58.             }
59.
60.             static void RegisterUser()
61.             {
62.                 Console.WriteLine("\n=== REGISTRASI USER ===");
63.
64.                 string username = "";
65.                 string password = "";
66.
```

```

67.         // Input dan validasi username
68.         while (true)
69.         {
70.             Console.Write("Masukkan Username (5-20 karakter, huruf dan angka saja): ");
71.             username = Console.ReadLine();
72.
73.             // Input Validation - Handling data invalid
74.             if (!User.IsValidUsername(username))
75.             {
76.                 Console.WriteLine("ERROR: Username tidak valid!");
77.                 Console.WriteLine("- Harus 5-20 karakter");
78.                 Console.WriteLine("- Hanya boleh huruf alfabet ASCII dan angka");
79.                 Console.WriteLine("Silakan coba lagi.\n");
80.                 continue;
81.             }
82.
83.             // Cek apakah username sudah ada
84.             if (users.Any(u => u.Username.Equals(username,
StringComparison.OrdinalIgnoreCase)))
85.             {
86.                 Console.WriteLine("ERROR: Username sudah terdaftar! Silakan gunakan
username lain.\n");
87.                 continue;
88.             }
89.
90.             break;
91.         }
92.
93.         // Input dan validasi password
94.         while (true)
95.         {
96.             Console.Write("Masukkan Password (8-30 karakter, harus ada huruf, angka,
dan !@#$$%^&*): ");
97.             password = ReadPassword();
98.
99.             // Input Validation - Handling data invalid
100.            if (!User.IsValidPassword(password))
101.            {
102.                Console.WriteLine("\nERROR: Password tidak valid!");
103.                Console.WriteLine("- Minimal 8 karakter, maksimal 30 karakter");
104.                Console.WriteLine("- Harus mengandung minimal 1 huruf");
105.                Console.WriteLine("- Harus mengandung minimal 1 angka");
106.                Console.WriteLine("- Harus mengandung minimal 1 karakter khusus
(!@#$$%^&*)");
107.                Console.WriteLine("Silakan coba lagi.\n");
108.                continue;
109.            }
110.
111.            // Password Management - Password rules
112.            if (User.PasswordContainsUsername(password, username))
113.            {
114.                Console.WriteLine("\nERROR: Password tidak boleh mengandung
username!");
115.                Console.WriteLine("Silakan coba lagi.\n");
116.                continue;
117.            }
118.
119.            break;
120.        }
121.
122.        // Password Management - Password hashing
123.        string hashedPassword = HashPassword(password);
124.
125.        // Simpan user baru
126.        User newUser = new User(username, hashedPassword);
127.        users.Add(newUser);
128.
129.        // Simpan ke file JSON
130.        SaveUsersToFile();

```

```

131.
132.         Console.WriteLine($"\\nSUKSES: User '{username}' berhasil didaftarkan!\\n");
133.     }
134.
135.     static void LoginUser()
136.     {
137.         Console.WriteLine("\\n=== LOGIN USER ===");
138.
139.         Console.Write("Username: ");
140.         string username = Console.ReadLine();
141.
142.         Console.Write("Password: ");
143.         string password = ReadPassword();
144.
145.         // Cari user berdasarkan username
146.         User user = users.FirstOrDefault(u => u.Username.Equals(username,
StringComparison.OrdinalIgnoreCase));
147.
148.         if (user == null)
149.         {
150.             Console.WriteLine("\\nERROR: Username tidak ditemukan!\\n");
151.             return;
152.         }
153.
154.         // Verifikasi password dengan hash
155.         string hashedInputPassword = HashPassword(password);
156.         if (user.PasswordHash != hashedInputPassword)
157.         {
158.             Console.WriteLine("\\nERROR: Password salah!\\n");
159.             return;
160.         }
161.
162.         Console.WriteLine($"\\nSUKSES: Login berhasil!");
163.         Console.WriteLine($"Selamat datang, {user.Username}!");
164.
165.         // Menu setelah login berhasil
166.         ShowUserMenu(user.Username);
167.     }
168.
169.     static void ShowAllUsers()
170.     {
171.         Console.WriteLine("\\n=== DAFTAR SEMUA USER ===");
172.
173.         if (users.Count == 0)
174.         {
175.             Console.WriteLine("Belum ada user terdaftar.\\n");
176.             return;
177.         }
178.
179.         Console.WriteLine($"Total user terdaftar: {users.Count}");
180.         Console.WriteLine("Username");
181.         Console.WriteLine("-----");
182.
183.         foreach (var user in users.OrderBy(u => u.Username))
184.         {
185.             Console.WriteLine($"{user.Username}");
186.         }
187.         Console.WriteLine();
188.     }
189.
190.     // Menu setelah user berhasil login
191.     static void ShowUserMenu(string username)
192.     {
193.         while (true)
194.         {
195.             Console.WriteLine($"\\n=== MENU USER: {username} ===");
196.             Console.WriteLine("1. Cek Kode Buah");
197.             Console.WriteLine("2. Game Posisi Karakter");
198.             Console.WriteLine("3. Logout");

```

```

199.         Console.Write("Pilih menu (1-3): ");
200.
201.         string choice = Console.ReadLine();
202.
203.         switch (choice)
204.         {
205.             case "1":
206.                 MenuKodeBuah();
207.                 break;
208.             case "2":
209.                 MenuGamePosisiKarakter();
210.                 break;
211.             case "3":
212.                 Console.WriteLine("Logout berhasil!\n");
213.                 return;
214.             default:
215.                 Console.WriteLine("Pilihan tidak valid! Silakan coba lagi.\n");
216.                 break;
217.         }
218.     }
219. }
220.
221. // Menu untuk fitur Kode Buah
222. static void MenuKodeBuah()
223. {
224.     Console.WriteLine("\n=== CEK KODE BUAH ===");
225.     Console.WriteLine("Daftar buah yang tersedia:");
226.     Console.WriteLine("Apel, Aprikot, Alpukat, Pisang, Paprika, Blackberry,");
227.     Console.WriteLine("Ceri, Kelapa, Jagung, Kurma, Durian, Anggur, Melon,
Semangka");
228.
229.     while (true)
230.     {
231.         Console.Write("\nMasukkan nama buah (atau 'exit' untuk kembali): ");
232.         string namaBuah = Console.ReadLine();
233.
234.         if (namaBuah.ToLower() == "exit")
235.             break;
236.
237.         if (string.IsNullOrEmpty(namaBuah))
238.         {
239.             Console.WriteLine("Nama buah tidak boleh kosong!");
240.             continue;
241.         }
242.
243.         string kode = KodeBuah.GetKodeBuah(namaBuah);
244.         Console.WriteLine($"Kode untuk '{namaBuah}': {kode}");
245.     }
246. }
247.
248. // Menu untuk Game Posisi Karakter
249. static void MenuGamePosisiKarakter()
250. {
251.     Console.WriteLine("\n=== GAME POSISI KARAKTER ===");
252.     Console.WriteLine("NIM: 2311104053");
253.
254.     PosisiKarakterGame game = new PosisiKarakterGame(2311104053);
255.
256.     Console.WriteLine("Kontrol Game:");
257.     Console.WriteLine("S = Arah Bawah");
258.     Console.WriteLine("W = Arah Atas");
259.     Console.WriteLine("X = Terbang");
260.     Console.WriteLine("J = Turun/Landing");
261.     Console.WriteLine("'exit' = Keluar dari game");
262.
263.     game.PrintState();
264.
265.     while (true)
266.     {

```



```

267.         Console.Write("\nTekan tombol (S/W/X/J) atau 'exit' untuk keluar: ");
268.         string input = Console.ReadLine();
269.
270.         if (input.ToLower() == "exit")
271.             break;
272.
273.         if (string.IsNullOrEmpty(input) || input.Length != 1)
274.         {
275.             Console.WriteLine("Input harus 1 karakter (S/W/X/J)!");
276.             continue;
277.         }
278.
279.         char tombol = char.ToUpper(input[0]);
280.         game.TekanTombol(tombol);
281.         game.PrintState();
282.     }
283. }
284.
285. // Password Management - Password hashing dengan SHA256
286. static string HashPassword(string password)
287. {
288.     using (SHA256 sha256Hash = SHA256.Create())
289.     {
290.         // Tambahkan salt untuk keamanan ekstra
291.         string saltedPassword = password + "SecureSalt2024!";
292.         byte[] bytes =
sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(saltedPassword));
293.
294.         StringBuilder builder = new StringBuilder();
295.         for (int i = 0; i < bytes.Length; i++)
296.         {
297.             builder.Append(bytes[i].ToString("x2"));
298.         }
299.         return builder.ToString();
300.     }
301. }
302.
303. // Utility untuk membaca password dengan hidden input
304. static string ReadPassword()
305. {
306.     string password = "";
307.     ConsoleKeyInfo keyInfo;
308.
309.     do
310.     {
311.         keyInfo = Console.ReadKey(true);
312.
313.         if (keyInfo.Key != ConsoleKey.Backspace && keyInfo.Key != ConsoleKey.Enter)
314.         {
315.             password += keyInfo.KeyChar;
316.             Console.Write("*");
317.         }
318.         else if (keyInfo.Key == ConsoleKey.Backspace && password.Length > 0)
319.         {
320.             password = password.Substring(0, password.Length - 1);
321.             Console.Write("\b \b");
322.         }
323.     }
324.     while (keyInfo.Key != ConsoleKey.Enter);
325.
326.     return password;
327. }
328.
329. // Load users dari file JSON (manual parsing)
330. static void LoadUsersFromFile()
331. {
332.     try
333.     {
334.         if (File.Exists(UsersFilePath))

```

```

335.         {
336.             string json = File.ReadAllText(UsersFilePath);
337.             if (!string.IsNullOrEmpty(json))
338.             {
339.                 users = ParseUsersFromJson(json);
340.                 Console.WriteLine($"Berhasil memuat {users.Count} user dari
file.\n");
341.             }
342.         }
343.         else
344.         {
345.             Console.WriteLine("File users.json tidak ditemukan. Membuat file
baru.\n");
346.         }
347.     }
348.     catch (Exception ex)
349.     {
350.         Console.WriteLine($"Error loading users: {ex.Message}");
351.         users = new List<User>();
352.     }
353. }
354.
355. // Simpan users ke file JSON (manual serialization)
356. static void SaveUsersToFile()
357. {
358.     try
359.     {
360.         string json = SerializeUsersToJson(users);
361.         File.WriteAllText(UsersFilePath, json);
362.         Console.WriteLine("Data user berhasil disimpan ke users.json");
363.     }
364.     catch (Exception ex)
365.     {
366.         Console.WriteLine($"Error saving users: {ex.Message}");
367.     }
368. }
369.
370. // Manual JSON parsing untuk User list
371. static List<User> ParseUsersFromJson(string json)
372. {
373.     List<User> userList = new List<User>();
374.
375.     try
376.     {
377.         // Hapus whitespace dan bracket
378.         json = json.Trim().Trim('[', ''];
379.
380.         if (string.IsNullOrEmpty(json))
381.             return userList;
382.
383.         // Split berdasarkan object separator
384.         string[] userObjects = json.Split(new string[] { "},{ " },
StringSplitOptions.RemoveEmptyEntries);
385.
386.         foreach (string userObj in userObjects)
387.         {
388.             // Clean up object brackets
389.             string cleanObj = userObj.Trim().Trim('{', '}');
390.
391.             string username = "";
392.             string passwordHash = "";
393.
394.             // Parse properties
395.             string[] properties = cleanObj.Split(',');
396.             foreach (string prop in properties)
397.             {
398.                 string[] keyValue = prop.Split(':');
399.                 if (keyValue.Length == 2)
400.                 {

```

```

401.         string key = keyValue[0].Trim().Trim(' ');
402.         string value = keyValue[1].Trim().Trim(' ');
403.
404.         if (key == "Username")
405.             username = value;
406.         else if (key == "PasswordHash")
407.             passwordHash = value;
408.     }
409. }
410.
411.         if (!string.IsNullOrEmpty(username) &&
!string.IsNullOrEmpty(passwordHash))
412.         {
413.             userList.Add(new User(username, passwordHash));
414.         }
415.     }
416. }
417. catch (Exception ex)
418. {
419.     Console.WriteLine($"Error parsing JSON: {ex.Message}");
420. }
421.
422.     return userList;
423. }
424.
425. // Manual JSON serialization untuk User list
426. static string SerializeUsersToJson(List<User> userList)
427. {
428.     StringBuilder sb = new StringBuilder();
429.     sb.AppendLine("[");
430.
431.     for (int i = 0; i < userList.Count; i++)
432.     {
433.         sb.AppendLine("    {");
434.         sb.AppendLine($"        \"Username\": \"{userList[i].Username}\"");
435.         sb.AppendLine($"        \"PasswordHash\": \"{userList[i].PasswordHash}\"");
436.
437.         if (i < userList.Count - 1)
438.             sb.AppendLine("    },");
439.         else
440.             sb.AppendLine("    }");
441.     }
442.
443.     sb.AppendLine("]");
444.     return sb.ToString();
445. }
446. }
447. }
448.

```

Penjelasan codingan:

- KodeBuah.cs: Berfungsi sebagai kamus sederhana untuk mengubah nama buah menjadi kode (misal: "Apel" -> "A00").
- PosisiKarakterGame.cs: Mengelola perubahan posisi (state) sebuah karakter game berdasarkan input tombol (W, S, dll).
- User.cs: Mendefinisikan data pengguna (Username, PasswordHash) dan berisi semua aturan untuk validasi username dan password.
- Program.cs: File utama yang menjalankan aplikasi, menampilkan menu, mengatur proses registrasi (dengan password hashing) & login, serta menghubungkan semua fitur lainnya.