

Nama : Naufal Aflakh Wijayanto

NIM : 2211104073

Kelas : SE063

A. Tujuan

Modul ini bertujuan untuk menerapkan prinsip *Secure Coding Practices* dalam pengembangan aplikasi desktop berbasis Python, terutama pada fitur **Registrasi dan Login**.

B. Fitur Aplikasi

1. Registrasi user (input username & password)
2. Penyimpanan user ke dalam file auth.json
3. Login user menggunakan data dari file tersebut
4. Implementasi Secure Coding:
 - Validasi input
 - Hash password
 - Password rules

C. Implementasi Secure Coding

1. Input Validation

Jenis Validasi	Implementasi
Validasi panjang	Username & password: minimal 8 karakter, maksimal 20
Validasi karakter	Username: hanya ASCII alphanumeric
Handling invalid	Ditolak dengan pesan yang jelas via <code>messagebox.showerror</code>

2. Password Management

Jenis	Implementasi
Password hashing	SHA256 digunakan untuk menyimpan password
Password rule	Password wajib memiliki angka dan karakter spesial
Anti reuse nama	Password tidak boleh mengandung username

Main.py

```
import tkinter as tk
from tkinter import messagebox
import json
import re
import hashlib
import os

# ==== Helper Functions ====

def hash_password(password):
    return hashlib.sha256(password.encode()).hexdigest()

def is_valid_username(username):
    return username.isalpha() and 5 <= len(username) <= 20

def is_valid_password(password, username):
    if len(password) < 8 or len(password) > 20:
        return False
    if username.lower() in password.lower():
        return False
    if not re.search(r"[!@#$%^&*]", password):
        return False
    return True

def load_users():
    if not os.path.exists('auth.json'):
        return {}
    with open("auth.json", "r") as file:
        try:
            return json.load(file)
        except json.JSONDecodeError:
            return {}

def save_users(users):
    with open("auth.json", "w") as file:
        json.dump(users, file)

# ==== UI and Logic ====

def register():
    username = entry_username.get()
    password = entry_password.get()
```

```

    if not is_valid_username(username):
        messagebox.showerror("Error", "Username harus 5-20 huruf alfabet ASCII.")
        return
    if not is_valid_password(password, username):
        messagebox.showerror("Error", "Password minimal 8 karakter, ada simbol unik, dan tidak
boleh mengandung username.")
        return

    users = load_users()
    if username in users:
        messagebox.showwarning("Warning", "Username sudah terdaftar.")
        return

    users[username] = hash_password(password)
    save_users(users)
    messagebox.showinfo("Success", "Registrasi berhasil!")

def login():
    username = entry_username.get()
    password = entry_password.get()

    users = load_users()
    if username not in users:
        messagebox.showerror("Error", "Username tidak ditemukan.")
        return

    if users[username] != hash_password(password):
        messagebox.showerror("Error", "Password salah.")
        return

    messagebox.showinfo("Success", f"Selamat datang, {username}!")

# ==== GUI ====

app = tk.Tk()
app.title("Login & Register - Modul 15")

tk.Label(app, text="Username").pack()
entry_username = tk.Entry(app)
entry_username.pack()

tk.Label(app, text="Password").pack()
entry_password = tk.Entry(app, show="*")

```

```
entry_password.pack()

tk.Button(app, text="Register", command=register).pack(pady=5)
tk.Button(app, text="Login", command=login).pack(pady=5)

app.mainloop()
```

Penjelasan kode:

Kode Python di atas merupakan sebuah aplikasi desktop sederhana berbasis GUI (Graphical User Interface) yang dibuat menggunakan library tkinter. Aplikasi ini memiliki dua fitur utama yaitu **registrasi** dan **login** dengan penerapan prinsip **secure coding**, seperti validasi input dan penyimpanan password secara aman menggunakan hashing SHA256.

Pada bagian awal kode, terdapat beberapa **fungsi bantu (helper functions)**. Fungsi `hash_password(password)` akan mengubah password menjadi *hash* menggunakan algoritma SHA256 agar tidak disimpan dalam bentuk asli (plaintext). Fungsi `is_valid_username(username)` memvalidasi username agar hanya berisi huruf (`isalpha()`) dan memiliki panjang antara 5 hingga 20 karakter. Sementara itu, `is_valid_password(password, username)` memastikan password minimal 8 karakter, maksimal 20 karakter, mengandung karakter spesial seperti `!@#$%^&*`, dan tidak boleh mengandung nama pengguna. Fungsi `load_users()` digunakan untuk membaca data pengguna dari file `auth.json`, dan menangani kemungkinan error jika file tidak ada atau rusak. Sedangkan `save_users(users)` berfungsi untuk menyimpan data pengguna dalam format JSON.

Selanjutnya, terdapat dua fungsi utama yaitu `register()` dan `login()`. Fungsi `register()` mengambil input dari user melalui `entry_username` dan `entry_password`, kemudian melakukan validasi menggunakan fungsi-fungsi sebelumnya. Jika validasi berhasil dan username belum pernah terdaftar, password di-*hash* dan disimpan ke file JSON. Jika tidak, akan ditampilkan pesan kesalahan menggunakan `messagebox`. Fungsi `login()` melakukan pengecekan apakah username dan hash dari password yang dimasukkan cocok dengan data di file `auth.json`. Jika cocok, login dianggap berhasil.

Terakhir, pada bagian GUI, dibuat sebuah jendela aplikasi menggunakan `tk.Tk()` dengan judul "Login & Register - Modul 15". Komponen input seperti label dan entry untuk username dan password ditambahkan, beserta dua tombol untuk Register dan Login yang terhubung dengan fungsi masing-masing. Program ini ditutup dengan `app.mainloop()` agar GUI tetap berjalan. Secara keseluruhan, aplikasi ini tidak hanya berfungsi dengan baik, tetapi juga memperhatikan aspek keamanan dasar sesuai dengan prinsip *Secure Coding Practices*.

Output

Login & Register - Modul 15

Username

Password

Register

Login

78°F Mostly cloudy 10:36 PM 6/11/2025

Login & Register - Modul 15

Username
nopal

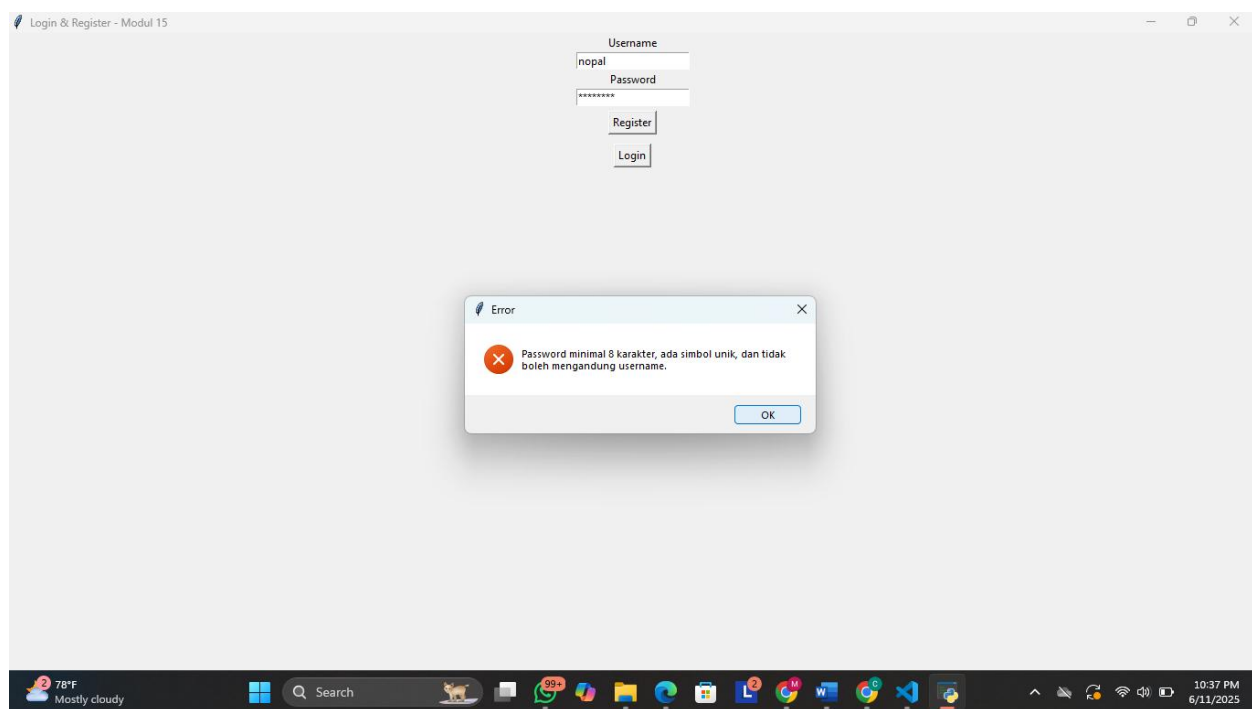
Password

Register

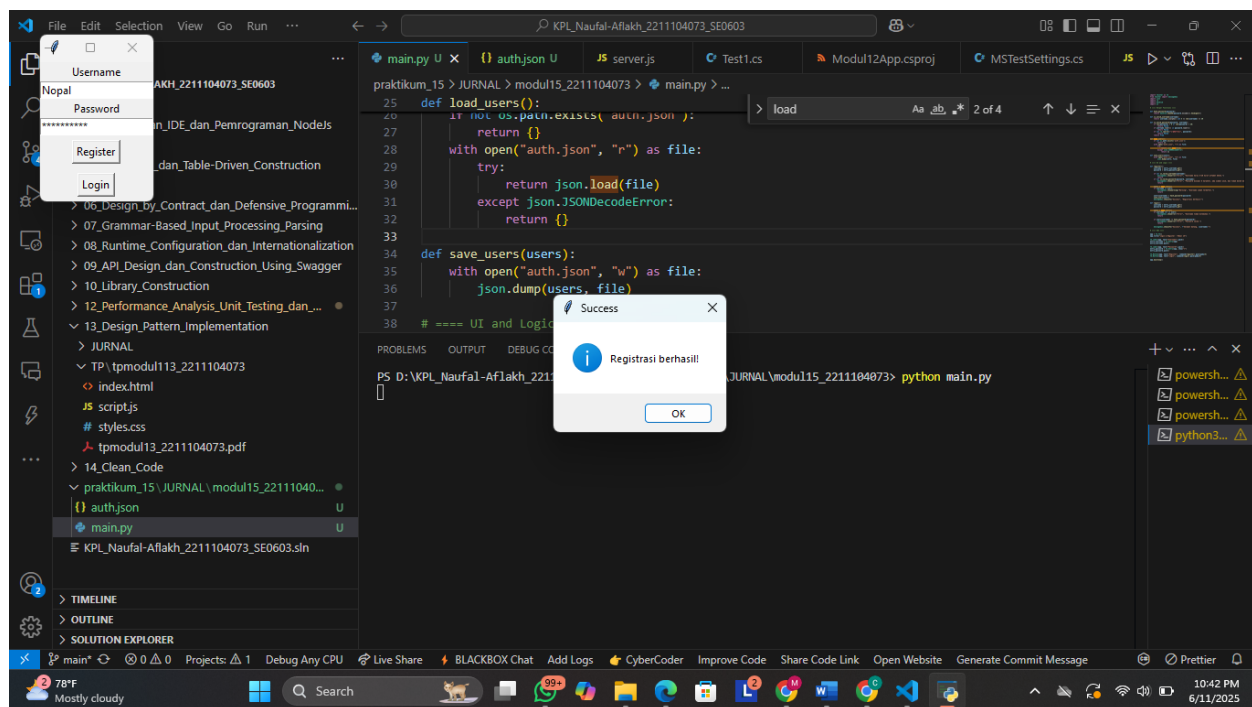
Login

78°F Mostly cloudy 10:37 PM 6/11/2025

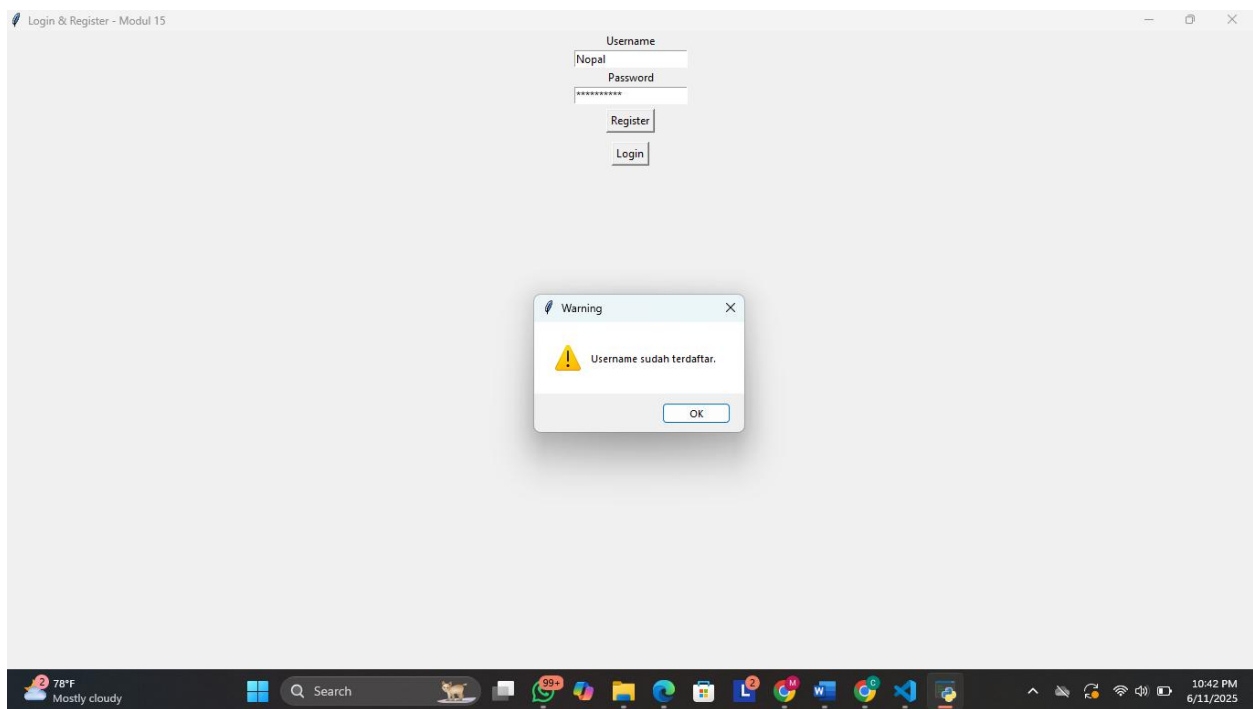
Menampilkan kesalahan jika tidak valid



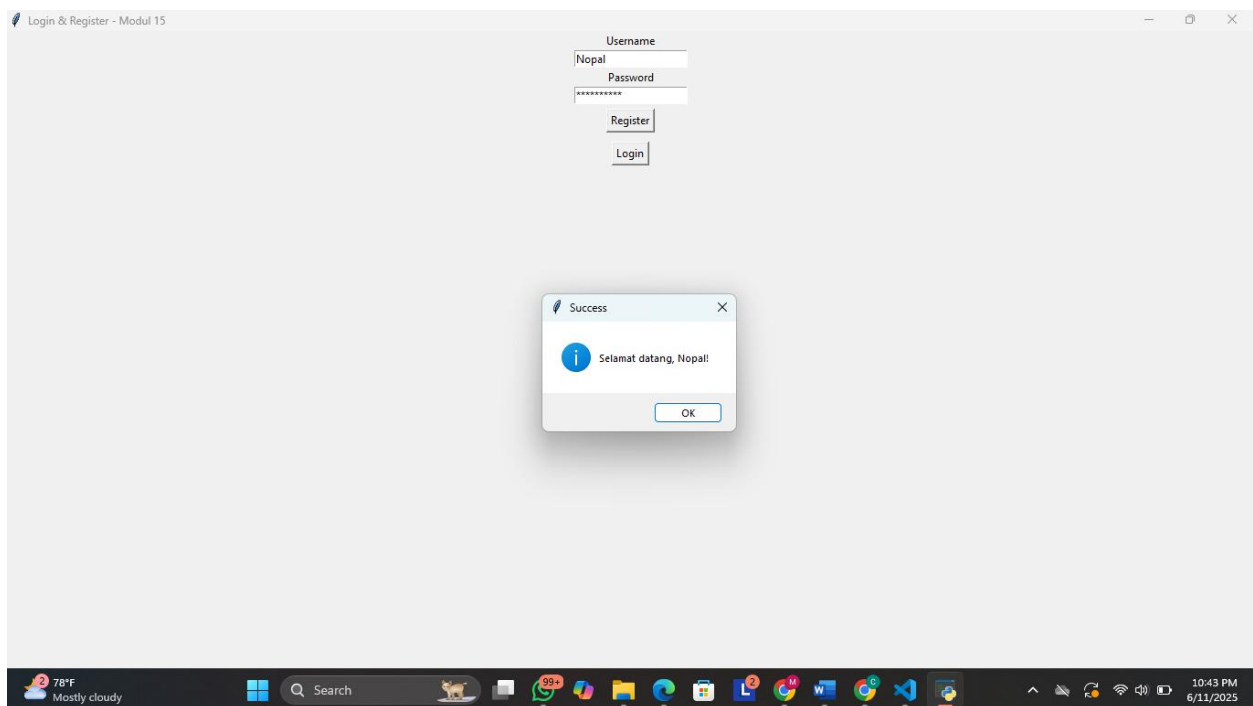
Menampilkan pesan jika sukses



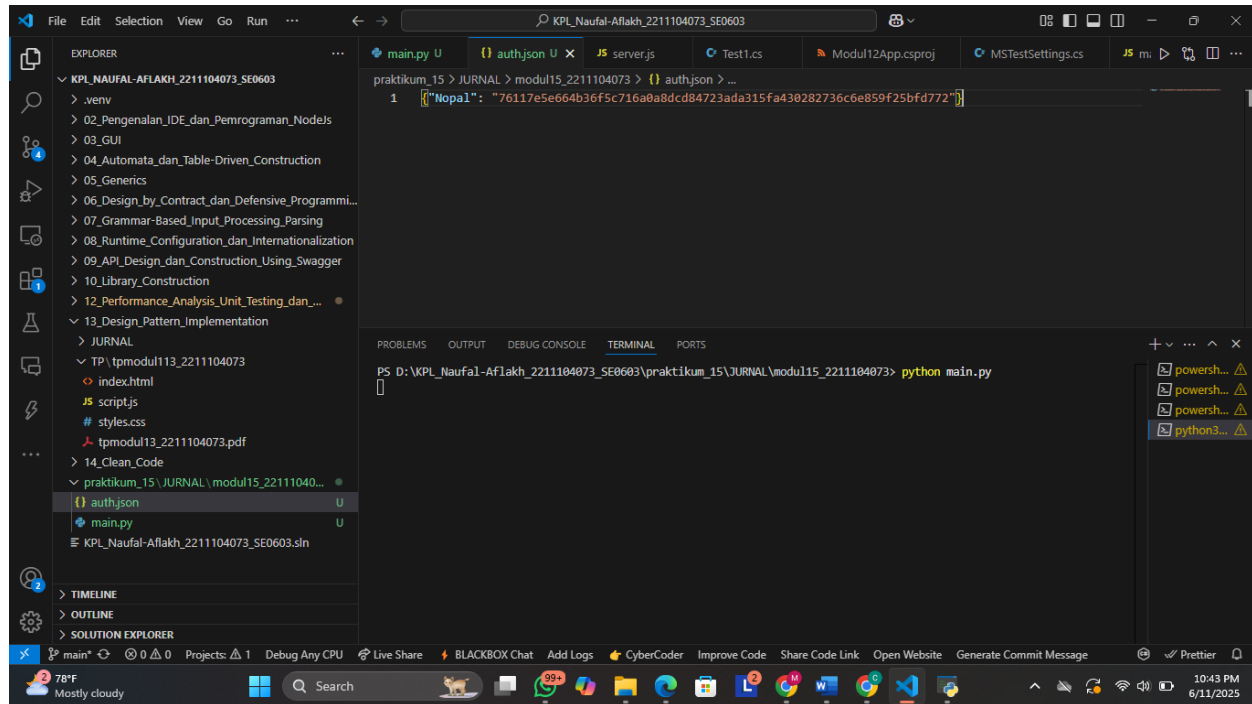
Menampilkan pesan jika registrasi lagi



Menampilkan pesan jika berhasil login



Hasil auth.json setelah berhasil registrasi



The screenshot shows a Visual Studio Code editor window. The Explorer sidebar on the left displays a project structure with folders like .env, 02_Pengenalan_JDE_dan_Pemrograman_Nodejs, 03_GUI, 04_Automata_dan_Table-Driven_Construction, 05_Generics, 06_Design_by_Contract_dan_Defensive_Programmi..., 07_Grammar-Based_Input_Processing_Parsing, 08_Runtime_Configuration_dan_Internationalization, 09_API_Design_dan_Construction_Using_Swagger, 10_Library_Construction, 12_Performance_Analysis_Unit_Testing_dan_..., 13_Design_Pattern_Implementation, and JURNAL. Under JURNAL, there is a folder TP\tpmodul113_2211104073 containing index.html, script.js, styles.css, and tpmodul13_2211104073.pdf. Another folder praktikum_15\JURNAL\modul15_2211104073 contains auth.json, main.py, and KPL_Naufal-Aflakh_2211104073_SE0603.sln. The main editor area shows the content of auth.json, which is a JSON object: {"Nopai": "76117e5e664b36f5c716a8a8dcd84723ada315fa430282736c6e859f25bfd772"}. The bottom panel shows a terminal window with the command PS D:\KPL_Naufal-Aflakh_2211104073\praktikum_15\JURNAL\modul15_2211104073> python main.py and the output of the command.

Kesimpulan

Pada Jurnal Modul 15 ini, telah berhasil dikembangkan sebuah aplikasi desktop sederhana menggunakan Python dan Tkinter dengan menerapkan prinsip **Secure Coding Practices**. Aplikasi ini terdiri dari dua fitur utama, yaitu **registrasi** dan **login** pengguna, yang datanya disimpan dalam format JSON. Dalam pengembangannya, beberapa aspek keamanan telah diimplementasikan dengan baik, antara lain:

1. Input Validation:

- Validasi **username** agar hanya terdiri dari huruf alfabet (ASCII) dan memiliki panjang antara 5 hingga 20 karakter.
- Validasi **password** agar memiliki panjang antara 8 hingga 20 karakter, mengandung minimal satu karakter unik (!@#\$%^&*), dan tidak mengandung kata dari username.

2. Password Management:

- Password pengguna tidak disimpan dalam bentuk asli, tetapi di-*hash* menggunakan algoritma **SHA256** sebelum disimpan ke dalam file auth.json. Hal

ini bertujuan untuk menjaga kerahasiaan password dan menghindari kebocoran data.

3. **Error Handling:**

- Sistem menangani kemungkinan error seperti kesalahan input dan file JSON yang kosong atau rusak dengan baik, sehingga tidak menimbulkan runtime error yang tidak dikendalikan.

Melalui penerapan prinsip-prinsip secure coding ini, aplikasi menjadi lebih aman terhadap berbagai risiko keamanan dasar seperti *password leak*, *input injection*, dan kesalahan validasi. Selain itu, penggunaan antarmuka grafis melalui tkinter menjadikan aplikasi ini mudah digunakan oleh pengguna umum. Dengan demikian, tugas ini tidak hanya berhasil secara fungsional, tetapi juga menunjukkan pemahaman yang baik dalam membangun aplikasi yang aman dan handal.