LAPORAN PRAKTIKUM

POSTTEST 5

ALGORITMA PEMROGRAMAN DASAR

Disusun oleh:

Muhammad Alfauzi Syahputra 2509106006

Kelas A1 ‘25

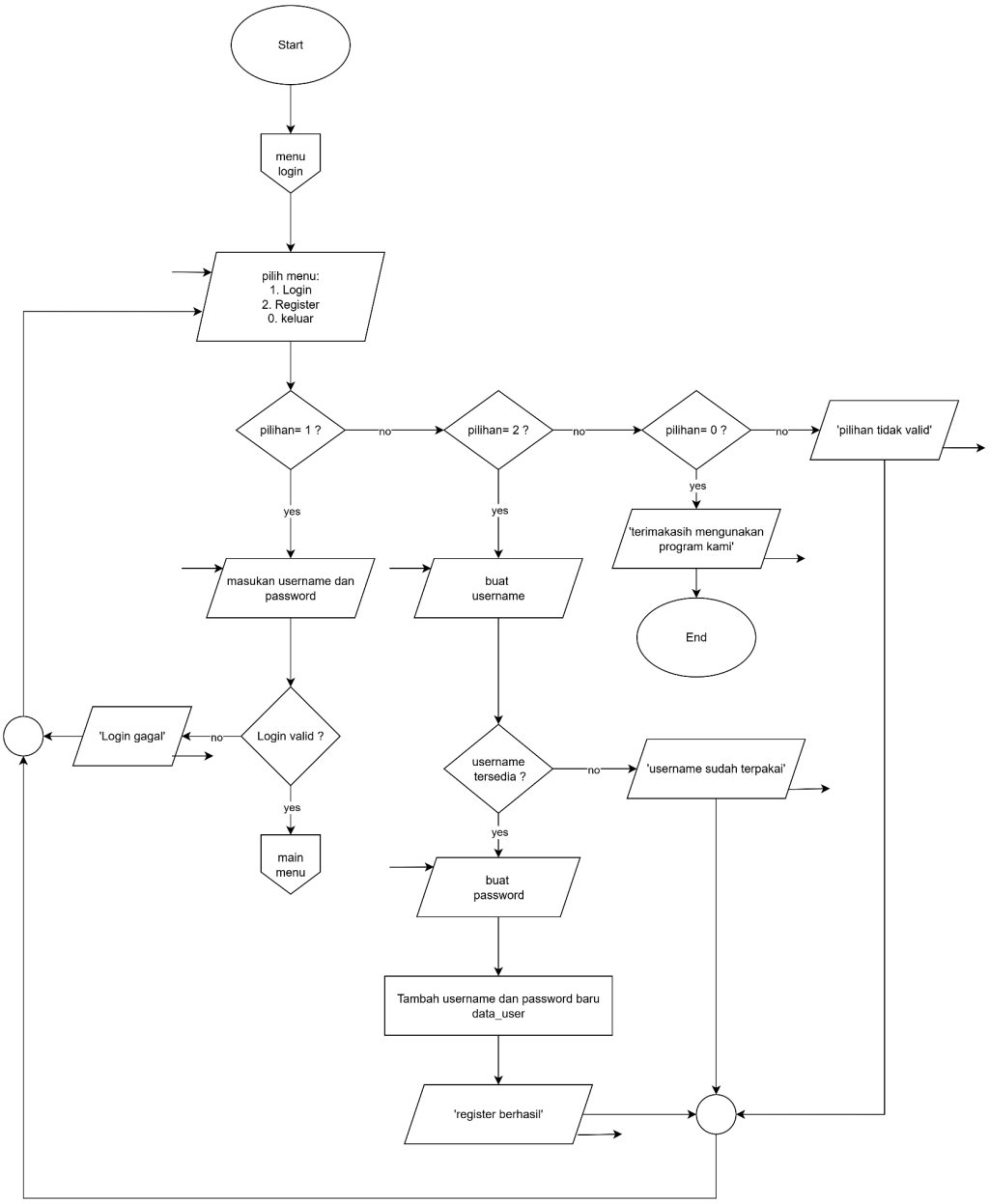
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

SAMARINDA

2025

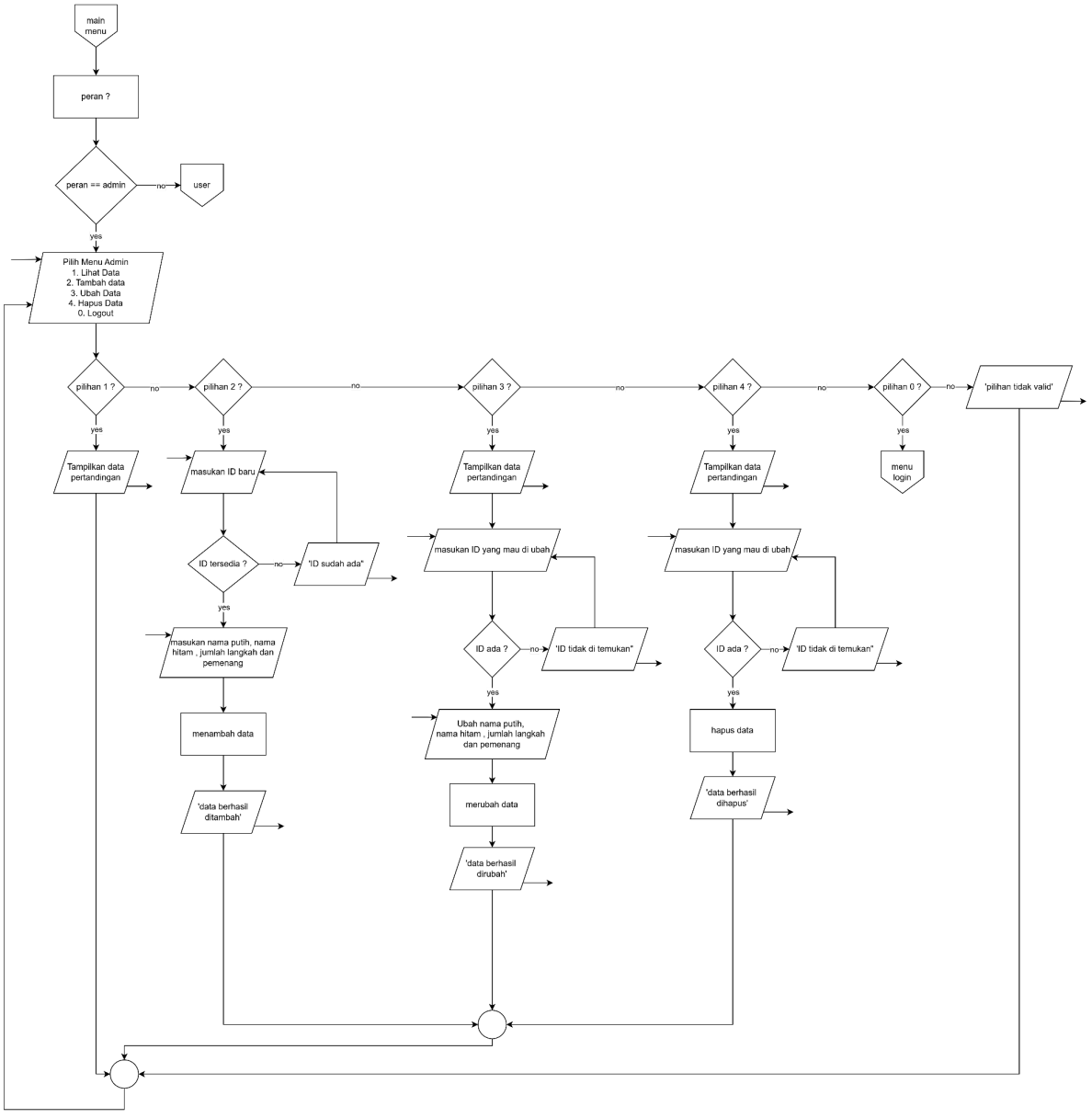
1. Flowchart



gambar 1.1 flowchart menu login dan register

Pada flowchart gambar 1.1, alur program dimulai dengan simbol Start, kemudian langsung ke proses OUTPUT di mana sistem menampilkan menu utama yang memberikan pilihan untuk Login, Register, atau Keluar. Setelah itu, sistem akan menunggu INPUT pilihan dari pengguna. Simbol Decision kemudian memeriksa pilihan tersebut. Jika pengguna memilih '1' (Login), alur berlanjut ke proses INPUT username dan password, yang kemudian divalidasi oleh Decision kedua. Jika valid, pengguna akan diarahkan ke menu sesuai perannya (admin atau user), namun jika tidak valid, program akan menampilkan pesan error. Jika pengguna memilih '2' (Register), alur akan mengarah ke proses registrasi sistem akan meminta INPUT username baru dan menggunakan Decision untuk memeriksa apakah username tersebut sudah ada. Jika sudah ada, program akan menampilkan pesan error, tetapi

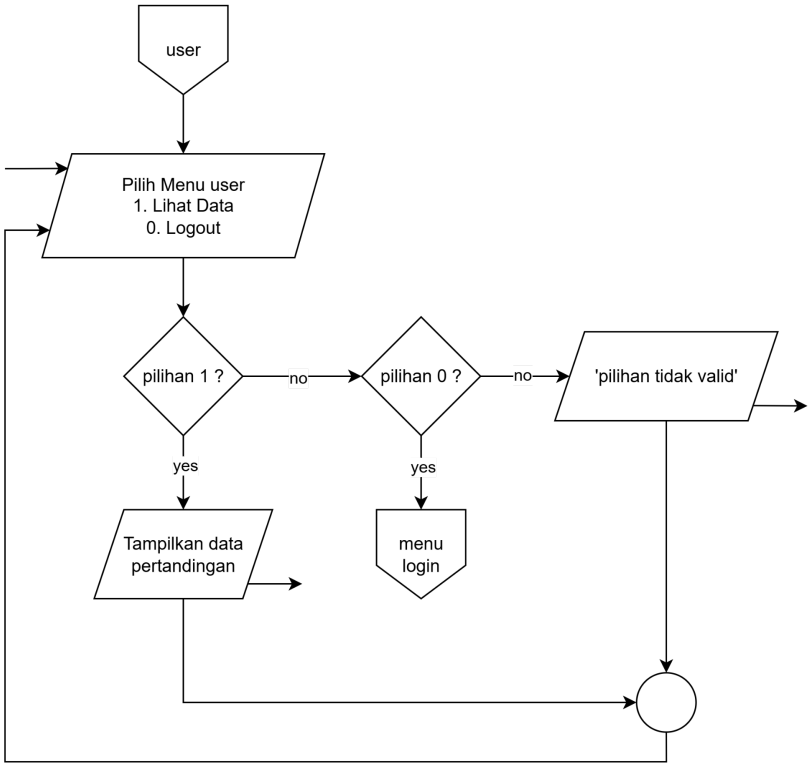
jika tersedia, program akan meminta INPUT password baru dan menambahkan data pengguna tersebut ke sistem dengan peran default sebagai 'user'. Terakhir, jika pengguna memilih '0' (Keluar), maka program akan berakhir sesuai dengan alur yang ditandai oleh simbol End.



gambar 1.2 flowchart menu admin

Lanjutan dari flowchart sebelumnya, flowchart pada gambar 1.2 menggambarkan proses setelah admin berhasil login. Alur dimulai dengan program melakukan OUTPUT menu admin yang berisi opsi CRUD (Create, Read, Update, Delete) dan Logout. Setelah menerima INPUT pilihan menu dari admin, sebuah simbol Decision akan memeriksa pilihan tersebut dan mengarahkan alur ke proses yang sesuai: pilihan '1' akan menjalankan proses Read untuk menampilkan semua data, '2' untuk proses Create data baru, '3' untuk proses Update data yang ada, dan '4' untuk proses Delete data. Jika admin memilih '0', alur akan

mengarah ke proses Logout untuk kembali ke menu utama. Flowchart juga menunjukkan bahwa setiap proses CRUD melibatkan validasi data dan setelah selesai, alurnya akan kembali lagi untuk menampilkan menu admin, menciptakan sebuah siklus operasi hingga admin memilih untuk logout.



gambar 1.3 flowchart menu user

Pada flowchart gambar 1.3, alur untuk pengguna dengan peran 'user' ditampilkan. Setelah login berhasil, program akan melakukan OUTPUT menu user yang lebih sederhana, hanya berisi opsi untuk melihat data dan Logout.

Setelah menerima INPUT dari pengguna, Decision akan memeriksa pilihan tersebut. Jika pengguna memilih '1', program akan menampilkan semua data pertandingan. Jika memilih '0', program akan mengakhiri sesi user dan kembali ke menu utama. Alur ini kemudian berakhir pada simbol End yang mengembalikan pengguna ke menu utama.

2. Deskripsi Singkat Program

Kode program ini dibuat untuk sistem manajemen "Pencatatan Hasil Pertandingan Catur". Program ini dirancang untuk melayani dua jenis pengguna, yaitu admin dan user biasa, dengan alur dan hak akses yang berbeda. Admin memiliki kontrol penuh untuk melakukan operasi Create, Read, Update, dan Delete (CRUD) terhadap data pertandingan. Sementara itu, user biasa hanya memiliki hak akses untuk melihat data yang ada (Read-only). Program ini juga dilengkapi dengan fitur registrasi untuk pengguna baru.

3. Source Code

A. Inisialisasi Data dan Tampilan Menu

Pada bagian awal program, dilakukan impor data dari file terpisah (*data.py* dan *menu.py*). File *data.py* berfungsi untuk menyimpan data awal pengguna dan data pertandingan dalam struktur *nested list*. File *menu.py* digunakan untuk mendefinisikan tampilan menu menggunakan library *prettytable* agar lebih terstruktur dan rapi.

Source Code data,py :

```
data_user = [
    ['alfauzi', '006', 'admin'],
    ['putra', '123', 'user'],
    ['admin', 'admin123', 'admin']
]

data_pertandingan = [
    [1, 'Magnus Carlsen', 'Hikaru Nakamura', 45, 'Putih Menang'],
    [2, 'Anish Giri', 'Fabiano Caruana', 60, 'Seri'],
    [3, 'Ian Nepomniachtchi', 'Ding Liren', 38, 'Hitam Menang']
]
```

Source Code menu,py :

```
from prettytable import PrettyTable

menu_utama = PrettyTable()
menu_utama.field_names = ['Pilihan', 'Deskripsi']
menu_utama.align['Deskripsi'] = 'l'
menu_utama.add_row(['1', 'Login                                 '])
menu_utama.add_row(['2', 'Register                              '])
menu_utama.add_row(['0', 'Keluar                                '])

menu_admin = PrettyTable()
menu_admin.field_names = ['Pilihan', 'Deskripsi']
menu_admin.align['Deskripsi'] = 'l'
```

```
menu_admin.add_row(['1', 'Lihat Data Pertandingan (Read)      '])
menu_admin.add_row(['2', 'Tambah Data Pertandingan (Create)   '])
menu_admin.add_row(['3', 'Ubah Data Pertandingan (Update)     '])
menu_admin.add_row(['4', 'Hapus Data Pertandingan (Delete)    '])
menu_admin.add_row(['0', 'Logout'])

menu_user = PrettyTable()
menu_user.field_names = ['Pilihan', 'Deskripsi']
menu_user.align['Deskripsi'] = 'l'
menu_user.add_row(['1', 'Lihat Data Pertandingan              '])
menu_user.add_row(['0', 'Logout                               '])
```

B. Fitur Menu Login, Registrasi dan keluar

Fitur ini adalah gerbang utama program yang berjalan dalam sebuah *while True* agar menu terus ditampilkan setelah setiap operasi selesai. Program akan meminta INPUT pilihan dari pengguna dan menggunakan *if-elif-else* untuk mengontrol alur. Jika pengguna memilih '1' (Login), program akan meminta username dan password, lalu menjalankan mesin validasi. Logikanya adalah dengan melakukan perulangan pada list *data_user* untuk memeriksa setiap data satu per satu. Di dalam perulangan, sebuah if statement akan memvalidasi apakah *user[0]* (username) dan *user[1]* (password) cocok dengan input pengguna. Jika kedua kondisi ini terpenuhi, status *login_berhasil* diubah menjadi *True*, peran pengguna disimpan dari *user[2]*, dan perulangan dihentikan dengan *break*. Jika pengguna memilih '2' (Register), program juga melakukan iterasi pada *data_user* untuk memastikan username belum terdaftar sebelum menambahkannya ke list menggunakan metode *.append()*. Terakhir, jika pengguna memilih '0' (Keluar), program akan menghentikan loop utama dengan perintah *break* dan program berakhir.

Source Code:

```
import os
from prettytable import PrettyTable
from colorama import Fore, Style, init
from data import data_pertandingan, data_user
from menu import menu_utama, menu_admin, menu_user
init(autoreset=True)

while True:
    os.system('cls || clear')
    print('PENCATATAN HASIL PERTANDINGAN CATUR')
    print(menu_utama)
    pilihan_awal = input('Masukkan pilihan Anda: ')
    login_berhasil = False

    if pilihan_awal == '1':
        os.system('cls || clear')
        print(Fore.YELLOW + ('HALAMAN LOGIN')
```

```
        input_username = input('Masukkan Username: ')
        input_password = input('Masukkan Password: ')
        for user in data_user:
            if user[0] == input_username and user[1] == input_password:
                peran_user = user[2]
                login_berhasil = True
                break
        if login_berhasil == True:
            print('Login berhasil')
            input('Tekan Enter untuk melanjutkan...' +

        else:
            print('Login gagal')
            input('Tekan Enter untuk kembali...')

    elif pilihan_awal == '2':
        os.system('cls || clear')
        username_baru = input('Masukkan Username Baru: ')
        username_sudah_ada = False
        for user in data_user:
            if user[0] == username_baru:
                username_sudah_ada = True
                break
        if username_sudah_ada == True:
            print('Username sudah digunakan.')
        else:
            password_baru = input('Masukkan Password Baru: ')
            data_user.append([username_baru, password_baru, 'user'])
            print('Registrasi berhasil! Silakan login.')
        input('Tekan Enter untuk kembali...')

    elif pilihan_awal == '0':
        print('Terima kasih telah menggunakan program kami!')
        break

    else:
        print('Pilihan tidak valid, silakan pilih 1, 2, atau 0.')
        input('Tekan Enter untuk mencoba lagi...')
```

C. Menu Admin READ

Fungsi ini bertugas untuk menampilkan seluruh data pertandingan yang tersimpan. Logikanya, program pertama-tama membuat sebuah objek tabel kosong dengan *tabel = PrettyTable()*, lalu mendefinisikan judul untuk setiap kolomnya menggunakan *tabel.field_names*. Setelah itu, program memeriksa apakah list *data_pertandingan* memiliki isi (*if data_pertandingan:*). Jika ya, program akan melakukan perulangan (*for...in*) untuk mengambil setiap data pertandingan satu per satu dan menambahkannya sebagai baris baru ke dalam tabel menggunakan *.add_row(pertandingan)*. Terakhir, setelah semua data ditambahkan, *print(tabel)* akan menampilkan tabel yang sudah terformat dengan rapi ke

layar. Jika list data ternyata kosong, program akan menampilkan pesan bahwa belum ada data.

Source Code:

```
            while True:
                os.system('cls || clear')

                if peran_user == 'admin':
                    print(f'Selamat Datang, {input_username}')
                    print(f'MENU ADMIN')
                    print(menu_admin)
                    pilihan_menu = input('Pilih menu: ')

                    if pilihan_menu == '1':
                        os.system('cls || clear')
                        print('DATA HASIL PERTANDINGAN')
                        tabel = PrettyTable()
                        tabel.field_names = ['ID', 'Pemain Putih', 'Pemain
Hitam', 'Jumlah Langkah', 'Pemenang']
                        if data_pertandingan:
                            for pertandingan in data_pertandingan:
                                tabel.add_row(pertandingan)
                            print(tabel)
                        else:
                            print('Belum ada data pertandingan.')
                        input('Tekan Enter untuk kembali...')
```

D. Menu Admin CREATE

Fungsi ini memungkinkan admin untuk menambah data baru, lengkap dengan validasi untuk memastikan integritas data. Prosesnya berjalan di dalam sebuah *while True* loop agar pengguna dapat mencoba lagi jika terjadi error. Pertama, program meminta INPUT ID dan melakukan validasi awal untuk memastikan inputnya adalah angka dengan *.isdigit()*. Jika valid, langkah validasi kedua adalah memastikan ID tersebut belum ada di dalam *data_pertandingan*. Logikanya, program mengonversi ID input menjadi integer *(int(input_id_str))*, lalu menginisialisasi variabel penanda *id_tersedia = True* dengan asumsi bahwa ID tersebut tersedia. Selanjutnya, program melakukan perulangan (*for i in data_pertandingan:*) untuk memeriksa setiap data yang sudah ada. Di dalam perulangan, *if i[0] == input_id:* akan membandingkan ID baru dengan ID dari setiap data. Jika ditemukan ID yang sama, variabel *id_tersedia* akan diubah menjadi *False* dan perulangan dihentikan dengan break. Jika setelah semua pengecekan *id_tersedia* tetap *True*, program akan melanjutkan untuk meminta input data lainnya. Setelah semua data terkumpul, metode *.append()* digunakan untuk menambahkan list baru ke dalam *data_pertandingan*, dan *break* akan menghentikan loop.

Source Code:

```
                    elif pilihan_menu == '2':
                        while True:
                            os.system('cls || clear')
                            print(('TAMBAH DATA PERTANDINGAN')
                            input_id_str = input('Masukkan ID Pertandingan (atau
ketik "X" untuk kembali): ')
                            if input_id_str == 'x':
                                break
                            elif not input_id_str.isdigit():
                                print('ID harus berupa angka!')
                                input('Tekan Enter untuk mencoba lagi...')
                                continue

                            input_id = int(input_id_str)
                            id_tersedia = True
                            for i in data_pertandingan:
                                if i[0] == input_id:
                                    id_tersedia = False
                                    break

                            if id_tersedia == True:
                                nama_putih = input('Nama Pemain Putih: ')
                                nama_hitam = input('Nama Pemain Hitam: ')
                                while True:
                                    jumlah_langkah_str = input('Jumlah Langkah
Total: ')

                                    if jumlah_langkah_str.isdigit():
                                        jumlah_langkah = int(jumlah_langkah_str)
                                        break
                                    else:
                                        print('Jumlah langkah harus berupa
angka!')
                                while True:
                                    print('Pilihan Hasil: 1. Putih Menang, 2.
Hitam Menang, 3. Seri')

                                    pilihan_hasil = input('Pilihan (1/2/3): ')
                                    if pilihan_hasil in ['1', '2', '3']:
                                        if pilihan_hasil == '1': hasil = 'Putih
Menang'

                                        elif pilihan_hasil == '2': hasil =
'Hitam Menang'

                                        else: hasil = 'Seri'
                                        break
                                    else:
                                        print('Pilihan tidak valid.')
                                data_pertandingan.append([input_id, nama_putih,
nama_hitam, jumlah_langkah, hasil])
                                print('Data berhasil ditambahkan!')
                                input('Tekan Enter untuk kembali...')
                                break
```

```
                            else:
                                print('ID sudah digunakan!')
                                input('Tekan Enter untuk mencoba lagi...')
```

E. Menu Admin UPDATE

Fungsi ini digunakan untuk mengubah data yang sudah ada berdasarkan ID yang dipilih. Prosesnya diawali dengan menampilkan seluruh data pertandingan agar pengguna tahu ID mana yang akan diubah. Selanjutnya, program masuk ke dalam sebuah loop *while True* yang akan terus meminta INPUT ID hingga ID yang valid dimasukkan. Di dalam loop ini, setelah memastikan input adalah angka, logika utamanya adalah mencari indeks (posisi) dari data yang akan diubah. Proses pencarian ini dimulai dengan mengonversi ID input menjadi integer dan menyiapkan variabel penanda *data_ditemukan = False*. Kemudian, program melakukan perulangan *for i in range(len(data_pertandingan))* untuk mendapatkan nomor indeks i dari setiap data. Jika ID cocok dengan *data_pertandingan[i][0]*, *data_ditemukan* diubah menjadi *True*, nomor indeks i disimpan, dan perulangan dihentikan dengan *break*. Jika data ditemukan, program akan meminta pengguna memasukkan semua informasi baru. Setelah semua input baru terkumpul, proses pembaruan data dilakukan oleh baris inti *data_pertandingan[indeks_data] = [input_id, nama_putih_baru, ...]*. Baris ini secara langsung mengganti (overwrite) elemen list lama pada posisi *indeks_data* dengan sebuah list baru yang berisi semua data yang baru saja dimasukkan pengguna. Setelah itu, loop dihentikan. Jika ID tidak ditemukan, program akan menampilkan pesan error dan meminta ulang ID.

Source Code:

```
                    elif pilihan_menu == '3':
                        os.system('cls || clear')
                        print('UBAH DATA PERTANDINGAN')
                        tabel = PrettyTable()
                        tabel.field_names = ['ID', 'Pemain Putih', 'Pemain
Hitam', 'Jumlah Langkah', 'Hasil']
                        if not data_pertandingan:
                            print('Belum ada data pertandingan untuk diubah.')
                            input('Tekan Enter untuk kembali...')
                            continue
                        for pertandingan in data_pertandingan:
                            tabel.add_row(pertandingan)
                        print(tabel)

                        while True:
                            input_id_str = input('Masukkan ID yang ingin diubah
(atau ketik "x" untuk kembali): ')
                            if input_id_str == 'x':
                                break
                            elif not input_id_str.isdigit():
```

```
                                print('Input tidak valid. ID harus berupa
angka.')
                                continue

                            input_id = int(input_id_str)
                            data_ditemukan = False

                            for i in range(len(data_pertandingan)):
                                if data_pertandingan[i][0] == input_id:
                                    data_ditemukan = True
                                    indeks_data = i
                                    break

                            if data_ditemukan == True:
                                print(f'Mengubah data untuk ID {input_id}:
{data_pertandingan[indeks_data]}')
                                nama_putih_baru = input('Nama Pemain Putih Baru:
')
                                nama_hitam_baru = input('Nama Pemain Hitam Baru:
')


                                while True:
                                    jumlah_langkah_baru_str = input('Jumlah
Langkah Baru: ')
                                    if jumlah_langkah_baru_str.isdigit():
                                        jumlah_langkah_baru =
int(jumlah_langkah_baru_str)
                                        break
                                    else:
                                        print('Input tidak valid, jumlah langkah
harus angka!')

                                while True:
                                    print('Pilihan Hasil Baru: 1. Putih Menang,
2. Hitam Menang, 3. Seri')
                                    pilihan_hasil_baru = input('Pilihan (1/2/3):
')
                                    if pilihan_hasil_baru in ['1', '2', '3']:
                                        if pilihan_hasil_baru == '1': hasil_baru
= 'Putih Menang'
                                        elif pilihan_hasil_baru == '2':
hasil_baru = 'Hitam Menang'
                                        else: hasil_baru = 'Seri'
                                        break
                                    else:
                                        print('Pilihan tidak valid.')

                                data_pertandingan[indeks_data] = [input_id,
nama_putih_baru, nama_hitam_baru, jumlah_langkah_baru, hasil_baru]
                                print('Data berhasil diubah!')
                                input('Tekan Enter untuk kembali...')
```

```
                                break
                            else:
                                print(f'ID {input_id} tidak ditemukan. Silakan
coba lagi.')
```

F. Menu Admin DELETE

Fungsi ini memungkinkan admin untuk menghapus data pertandingan berdasarkan ID. Mirip dengan proses Update, alurnya diawali dengan menampilkan seluruh data yang ada agar pengguna tahu ID mana yang akan dihapus. Program kemudian masuk ke dalam loop *while True* untuk meminta INPUT ID. Setelah memvalidasi bahwa input adalah angka, logika utamanya adalah mencari indeks dari data yang akan dihapus menggunakan perulangan *for i in range(len(data_pertandingan))*. Jika ID yang dimasukkan cocok dengan *data_pertandingan[i][0],* program akan menyimpan nomor indeks tersebut, mengubah penanda *data_ditemukan* menjadi *True,* dan menghentikan pencarian dengan *break.* Jika data ditemukan, inti dari proses penghapusan dieksekusi oleh baris *data_pertandingan.pop(indeks_data)*. Metode *.pop()* ini secara spesifik berfungsi untuk menghapus elemen dari list *data_pertandingan* pada posisi *indeks_data* yang sudah ditemukan. Setelah data berhasil dihapus, program akan memberikan notifikasi dan menghentikan loop. Jika ID tidak ditemukan, program akan menampilkan pesan error dan meminta ulang ID

Source Code:

```
                elif pilihan_menu == '4':
                    os.system('cls || clear')
                    print('HAPUS DATA PERTANDINGAN')
                    tabel = PrettyTable()
                    tabel.field_names = ['ID', 'Pemain Putih', 'Pemain
Hitam', 'Jumlah Langkah', 'Hasil']
                    if not data pertandingan:
                        print('Belum ada data pertandingan untuk diubah.')
                        input('Tekan Enter untuk kembali...')
                        continue
                    for pertandingan in data_pertandingan:
                        tabel.add_row(pertandingan)
                    print(tabel)

                    while True:
                        input_id_str = input('Masukkan ID yang mau dihapus
(atau ketik "X" untuk kembali): ')
                        if input_id_str == 'x':
                            break
                        elif not input_id_str.isdigit():
                            print('\nID harus berupa angka!')
                            input('Tekan Enter untuk mencoba lagi...')
                            continue
```

```
                input_id = int(input_id_str)
                data_ditemukan = False

                for i in range(len(data_pertandingan)):
                    if data_pertandingan[i][0] == input_id:
                        data_ditemukan = True
                        indeks_data = i
                        break

                if data_ditemukan == True:
                    data_pertandingan.pop(indeks_data)
                    print(f'Data ID {input_id} berhasil dihapus!')
                    input('Tekan Enter untuk kembali...')
                    break
                else:
                    print('ID tidak ditemukan.')
```

G. Menu Admin LOGOUT

Bagian terakhir dari menu admin adalah *elif pilihan_menu == '0'*, yang berfungsi sebagai fitur Logout. Logikanya sangat sederhana: jika pengguna memilih '0', program akan menampilkan pesan konfirmasi "Anda telah logout." Perintah *input()* digunakan untuk menjeda program sejenak, memberikan kesempatan bagi pengguna untuk membaca pesan tersebut. Setelah pengguna menekan Enter, perintah *break* akan dieksekusi. Perintah *break* ini adalah inti dari fungsionalitas logout, karena ia secara paksa menghentikan perulangan (loop) menu admin, sehingga program akan keluar dari sesi admin dan kembali ke menu utama (Login dan Register). Terakhir, blok *else* berfungsi sebagai penanganan error untuk semua input lain yang tidak valid (selain 1, 2, 3, 4, atau 0), di mana program akan menampilkan pesan "Pilihan tidak valid" dan meminta pengguna untuk mencoba lagi.

Source Code:

```
            elif pilihan_menu == '0':
                print('Anda telah logout.')
                input('Tekan Enter untuk kembali...')
                break
            else:
                print('Pilihan tidak valid.')
                input('Tekan Enter untuk mencoba lagi...')
```

H. Menu User

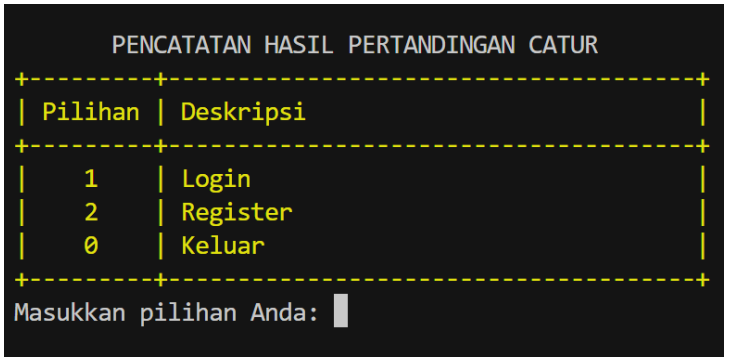
Setelah login berhasil, program memasuki sebuah loop *while True* yang akan terus berjalan hingga pengguna memilih logout. Di dalam loop ini, program menggunakan struktur kondisional *if peran_user == 'admin':*. Karena dalam sistem ini hanya ada dua peran, blok *else:* secara efektif akan dieksekusi untuk semua pengguna yang bukan admin, yaitu user

biasa. Blok ini pertama-tama akan menampilkan pesan selamat datang yang dipersonalisasi dan menu user yang lebih terbatas. Setelah menerima INPUT *pilihan_menu,* program akan menggunakan *if-elif-else* untuk menentukan tindakan: jika pengguna memilih '1', fungsionalitas untuk melihat data akan dijalankan, yang logikanya sama persis dengan menu *Read* pada admin. Jika pengguna memilih '0', fitur Logout dieksekusi; program akan menampilkan pesan konfirmasi, lalu perintah *break* akan menghentikan loop menu user, yang secara efektif mengembalikan pengguna ke menu utama. Terakhir, blok *else* di dalamnya berfungsi untuk menangani semua input lain yang tidak valid, dengan menampilkan pesan error.

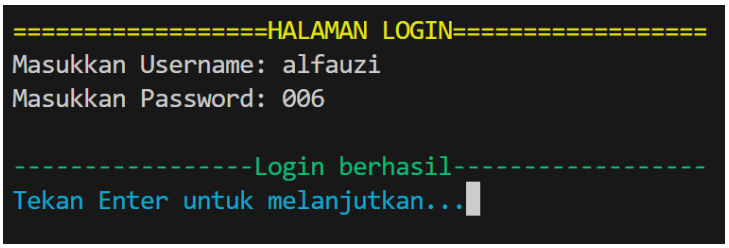
Source Code:

```
                else:
                    print(f'Selamat Datang, {input username}')
                    print(f'MENU USER')
                    print(menu_user)
                    pilihan_menu = input('Pilih menu: ')

                    if pilihan_menu == '1':
                        os.system('cls || clear')
                        print('DATA HASIL PERTANDINGAN')
                        tabel = PrettyTable()
                        tabel.field_names = ['ID', 'Pemain Putih', 'Pemain
Hitam', 'Jumlah Langkah', 'Pemenang']
                        if not data_pertandingan:
                            print('Belum ada data pertandingan.')
                        else:
                            for pertandingan in data_pertandingan:
                                tabel.add_row(pertandingan)
                            print(tabel)
                        input('Tekan Enter untuk kembali...')

                    elif pilihan_menu == '0':
                        print('Anda telah logout.')
                        input('Tekan Enter untuk kembali...')
                        break
                    else:
                        print('Pilihan tidak valid.')
                        input('Tekan Enter untuk mencoba lagi...')
```

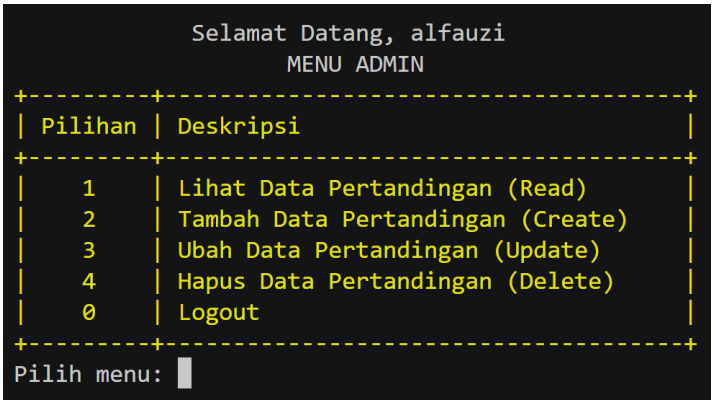
4. Hasil Output



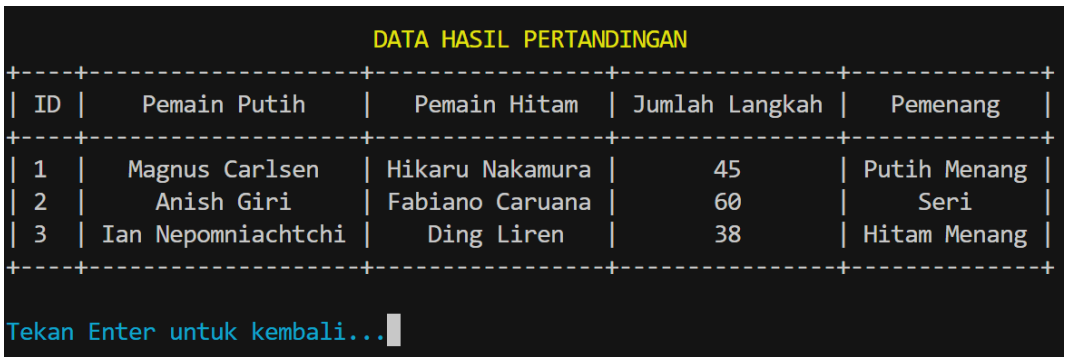
gambar 4.1 terminal menu login



gambar 4.2 terminal halaman login



gambar 4.3 terminal menu admin



gambar 4.4 terminal menu admin READ

```
==========TAMBAH DATA PERTANDINGAN==========
Masukkan ID Pertandingan (atau ketik "X" untuk kembali): 1

ID sudah digunakan!
Tekan Enter untuk mencoba lagi...
```

gambar 4.5 terminal menu admin CREATE ID sudah ada

```
==========TAMBAH DATA PERTANDINGAN==========
Masukkan ID Pertandingan (atau ketik "X" untuk kembali): dddddd

ID harus berupa angka!
Tekan Enter untuk mencoba lagi...
```

gambar 4.6 terminal menu admin CREATE input harus angka

```
==========TAMBAH DATA PERTANDINGAN==========
Masukkan ID Pertandingan (atau ketik "X" untuk kembali): 4
Nama Pemain Putih: Paul Morphy
Nama Pemain Hitam: Tigran Petrosian
Jumlah Langkah Total: 53

Pilihan Hasil: 1. Putih Menang, 2. Hitam Menang, 3. Seri
Pilihan (1/2/3): 2

Data berhasil ditambahkan!
Tekan Enter untuk kembali...
```

gambar 4.7 terminal menu admin CREATE berhasil di tambah

```
                        UBAH DATA PERTANDINGAN
+----+--------------------+------------------+----------------+--------------+
| ID |    Pemain Putih    |   Pemain Hitam   | Jumlah Langkah |    Hasil     |
+----+--------------------+------------------+----------------+--------------+
| 1  |   Magnus Carlsen   | Hikaru Nakamura  |       45       | Putih Menang |
| 2  |     Anish Giri     | Fabiano Caruana  |       60       |     Seri     |
| 3  | Ian Nepomniachtchi |    Ding Liren    |       38       | Hitam Menang |
| 4  |    Paul Morphy     | Tigran Petrosian |       53       | Hitam Menang |
+----+--------------------+------------------+----------------+--------------+

Masukkan ID yang ingin diubah (atau ketik "x" untuk kembali): a
Input tidak valid. ID harus berupa angka.

Masukkan ID yang ingin diubah (atau ketik "x" untuk kembali): 5
ID 5 tidak ditemukan. Silakan coba lagi.
```

gambar 4.8 terminal menu admin UPDATE input harus angka dan ID sudah ada

```
Masukkan ID yang ingin diubah (atau ketik "x" untuk kembali): 4

Mengubah data untuk ID 4: [4, 'Paul Morphy', 'Tigran Petrosian', 53, 'Hitam Menang']
Nama Pemain Putih Baru: Tigran Petrosian
Nama Pemain Hitam Baru: Paul Morphy
Jumlah Langkah Baru: 58

Pilihan Hasil Baru: 1. Putih Menang, 2. Hitam Menang, 3. Seri
Pilihan (1/2/3): 1

Data berhasil diubah!
Tekan Enter untuk kembali...
```

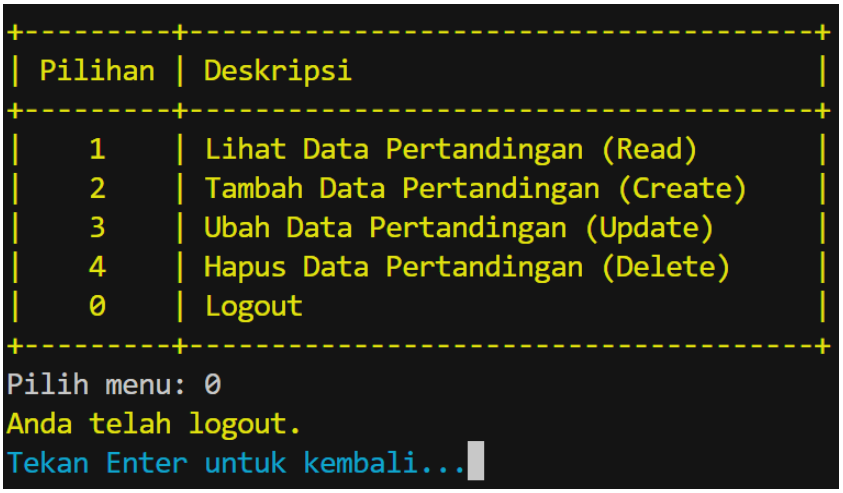
gambar 4.9 terminal menu admin CREATE berhasil diubah

```
                    HAPUS DATA PERTANDINGAN
+-----+--------------------+----------------+---------------+--------------+
|  ID |    Pemain Putih    |   Pemain Hitam | Jumlah Langkah |    Hasil     |
+-----+--------------------+----------------+---------------+--------------+
|  1  |   Magnus Carlsen   | Hikaru Nakamura |       45       | Putih Menang |
|  2  |     Anish Giri     | Fabiano Caruana |       60       |     Seri     |
|  3  | Ian Nepomniachtchi |    Ding Liren   |       38       | Hitam Menang |
|  4  |  Tigran Petrosian  |   Paul Morphy   |       58       | Putih Menang |
| 999 |     muheheheh      |    mahohohoh    |       69       |     Seri     |
+-----+--------------------+----------------+---------------+--------------+
Masukkan ID yang mau dihapus (atau ketik "X" untuk kembali): d

ID harus berupa angka!
Masukkan ID yang mau dihapus (atau ketik "X" untuk kembali): 5

ID tidak ditemukan.
Masukkan ID yang mau dihapus (atau ketik "X" untuk kembali): 999

Data ID 999 berhasil dihapus!
Tekan Enter untuk kembali...
```

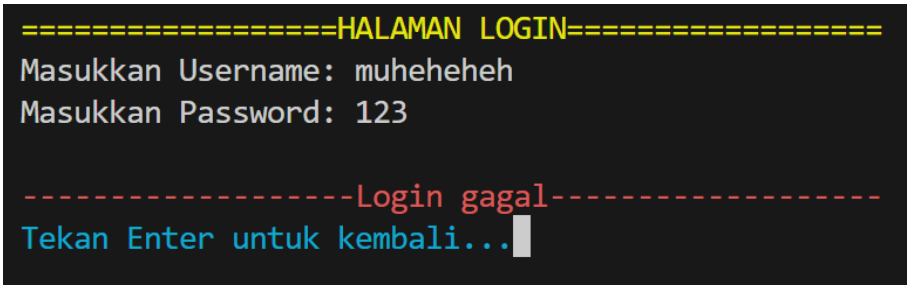
gambar 4.10 terminal menu admin DELETE harus angka ,ID tidak ditemukan dan ID999 berhasil dihapus

```
                    DATA HASIL PERTANDINGAN
+----+--------------------+----------------+---------------+--------------+
| ID |    Pemain Putih    |   Pemain Hitam | Jumlah Langkah |   Pemenang   |
+----+--------------------+----------------+---------------+--------------+
| 1  |   Magnus Carlsen   | Hikaru Nakamura |       45       | Putih Menang |
| 2  |     Anish Giri     | Fabiano Caruana |       60       |     Seri     |
| 3  | Ian Nepomniachtchi |    Ding Liren   |       38       | Hitam Menang |
| 4  |  Tigran Petrosian  |   Paul Morphy   |       58       | Putih Menang |
+----+--------------------+----------------+---------------+--------------+

Tekan Enter untuk kembali...
```

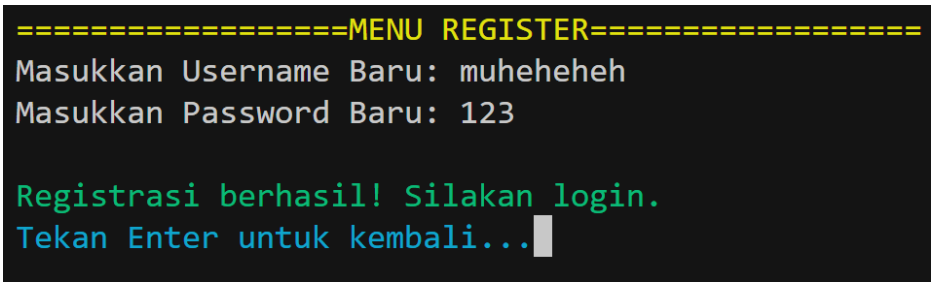
gambar 4.11 terminal menu admin READ setelah CURD



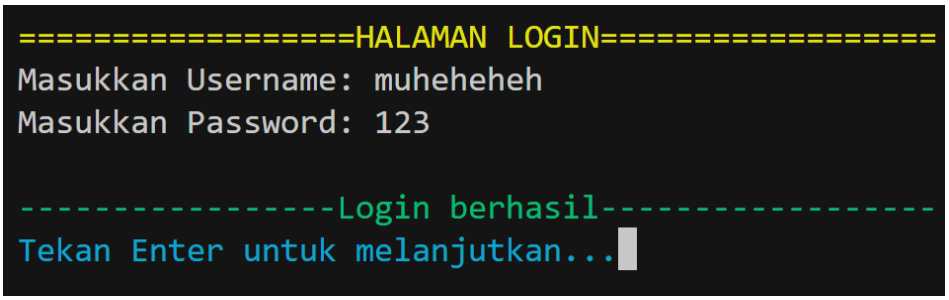
gambar 4.12 terminal menu admin LOGOUT



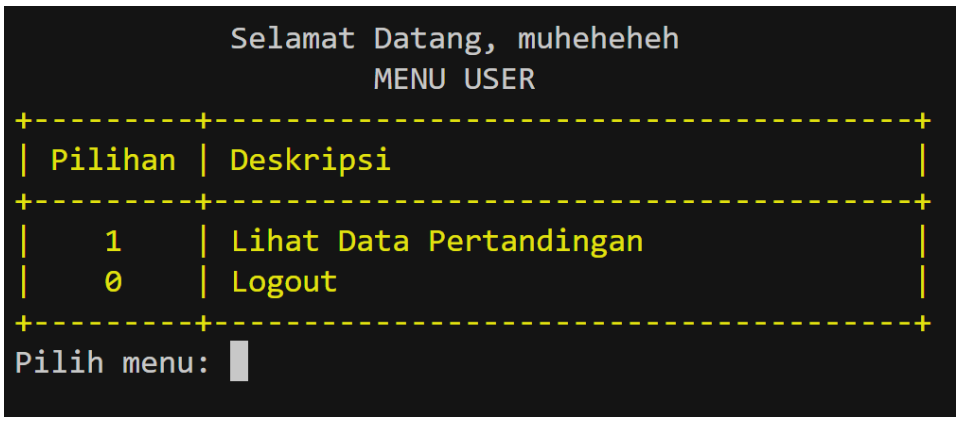
gambar 4.13 terminal halaman login gagal



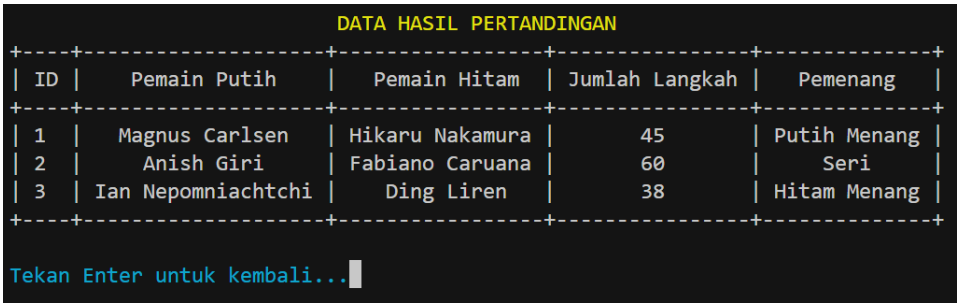
gambar 4.14 terminal menu register dan berhasil register



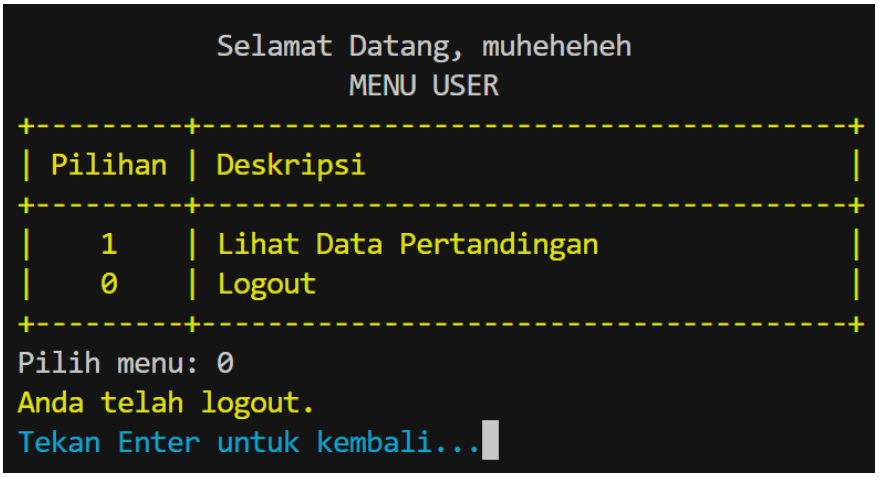
gambar 4.15 terminal halaman login dan berhasil



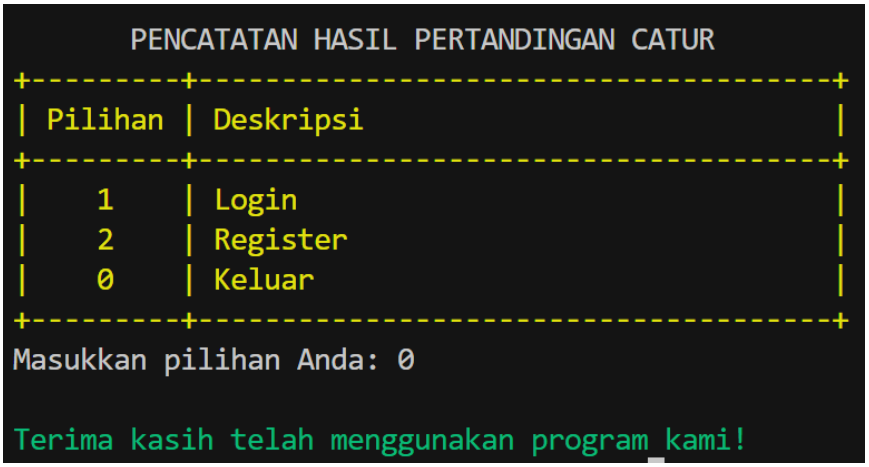
gambar 4.16 terminal menu user



gambar 4.17 terminal menu user READ



gambar 4.18 terminal menu LOGOUT



gambar 4.19 terminal menu keluar

```
                    DATA HASIL PERTANDINGAN

Belum ada data pertandingan.

Tekan Enter untuk kembali...
```

gambar 4.20 terminal READ belum ada data pertandingan

5. Langkah-langkah GIT

5.1 GIT Add

git add berfungsi untuk menambahkan file ke staging area, menandai file mana saja yang akan disimpan pada commit berikutnya.

```
PS D:\Kuliah\praktikum\praktikum-apd> git add .
PS D:\Kuliah\praktikum\praktikum-apd> git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   post-test/post-test-apd-5/2509106006-Muhammad-Alfauzi-Syahputra-PT-5.py
        new file:   post-test/post-test-apd-5/data.py
        new file:   post-test/post-test-apd-5/menu.py
```

gambar 5.1 menambahkan

5.2 GIT Commit

git commit digunakan untuk menyimpan perubahan yang sudah di-add ke repository lokal, biasanya disertai pesan singkat yang menjelaskan perubahan tersebut.

```
PS D:\Kuliah\praktikum\praktikum-apd> git commit -m 'posttest 5'
[main cbc5c28] posttest 5
 3 files changed, 294 insertions(+)
 create mode 100644 post-test/post-test-apd-5/2509106006-Muhammad-Alfauzi-Syahputra-PT-5.py
 create mode 100644 post-test/post-test-apd-5/data.py
 create mode 100644 post-test/post-test-apd-5/menu.py
```

gambar 5.2 menyimpan

5.3 GIT Push

git push digunakan untuk mengirim commit dari repository lokal ke repository remote di GitHub, agar perubahan bisa terlihat online dan diakses oleh orang lain.

```
PS D:\Kuliah\praktikum\praktikum-apd> git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 16 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 2.93 KiB | 2.93 MiB/s, done.
Total 7 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Alfauzi-S/praktikum-apd.git
   c2de9d8..cbc5c28  main -> main
PS D:\Kuliah\praktikum\praktikum-apd>
```

gambar 5.3 mendorong commit dari repository lokal ke repository remote