LAPORAN PRAKTIKUM

POSTTEST (8)

ALGORITMA PEMROGRAMAN DASAR

Disusun oleh:

Ferly Ahmad Nabil (2509106024)

Kelas (A2 ‘25)

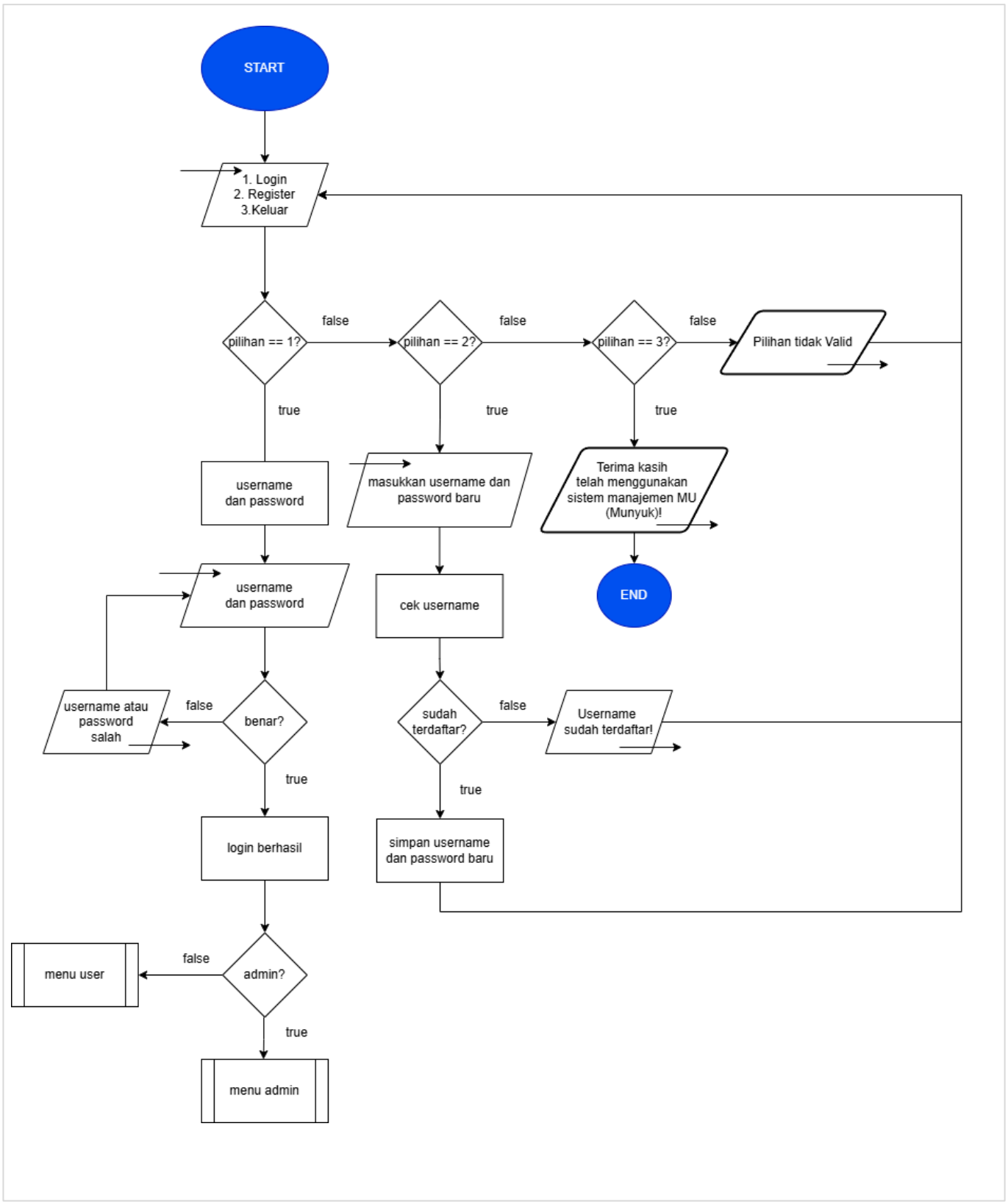
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

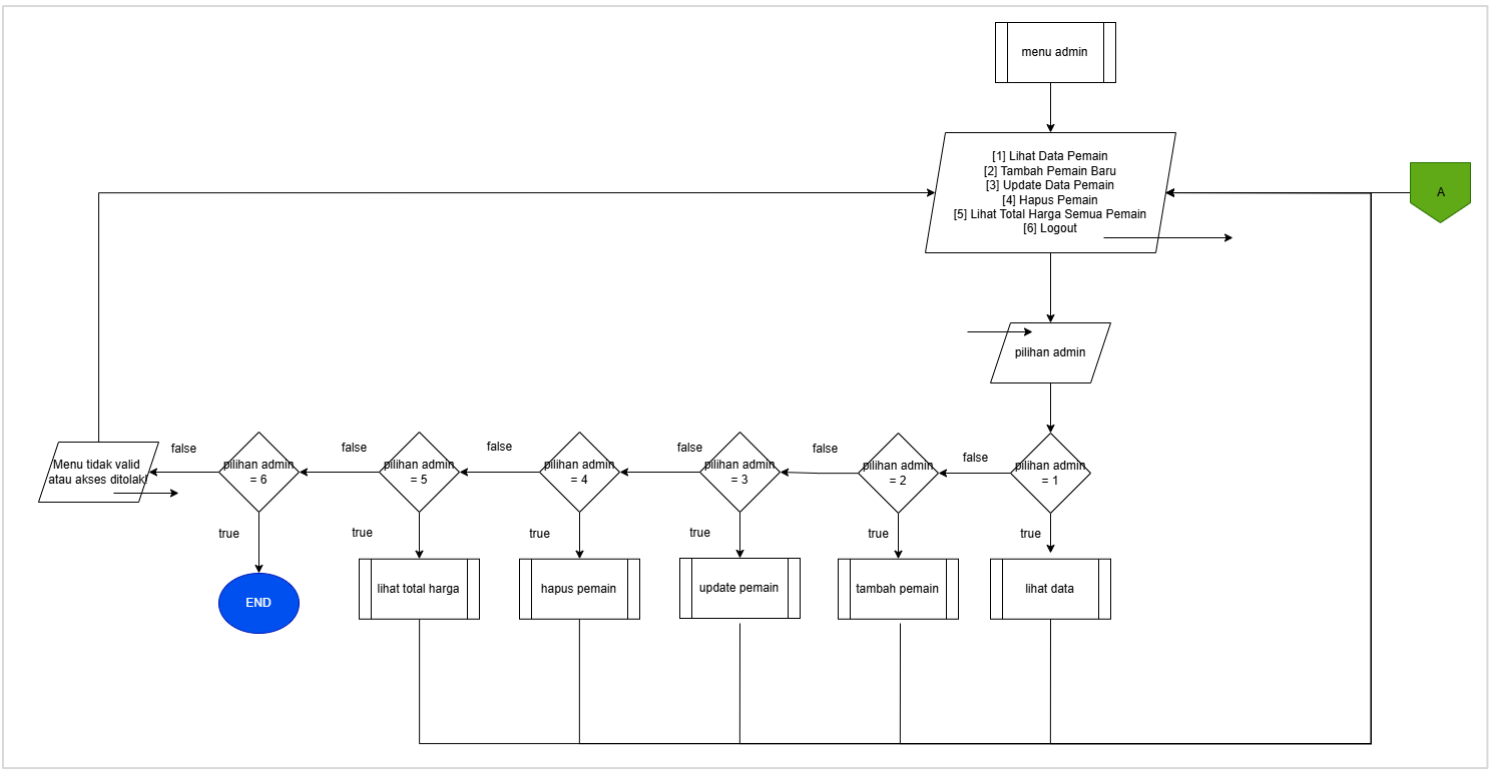
SAMARINDA

2025

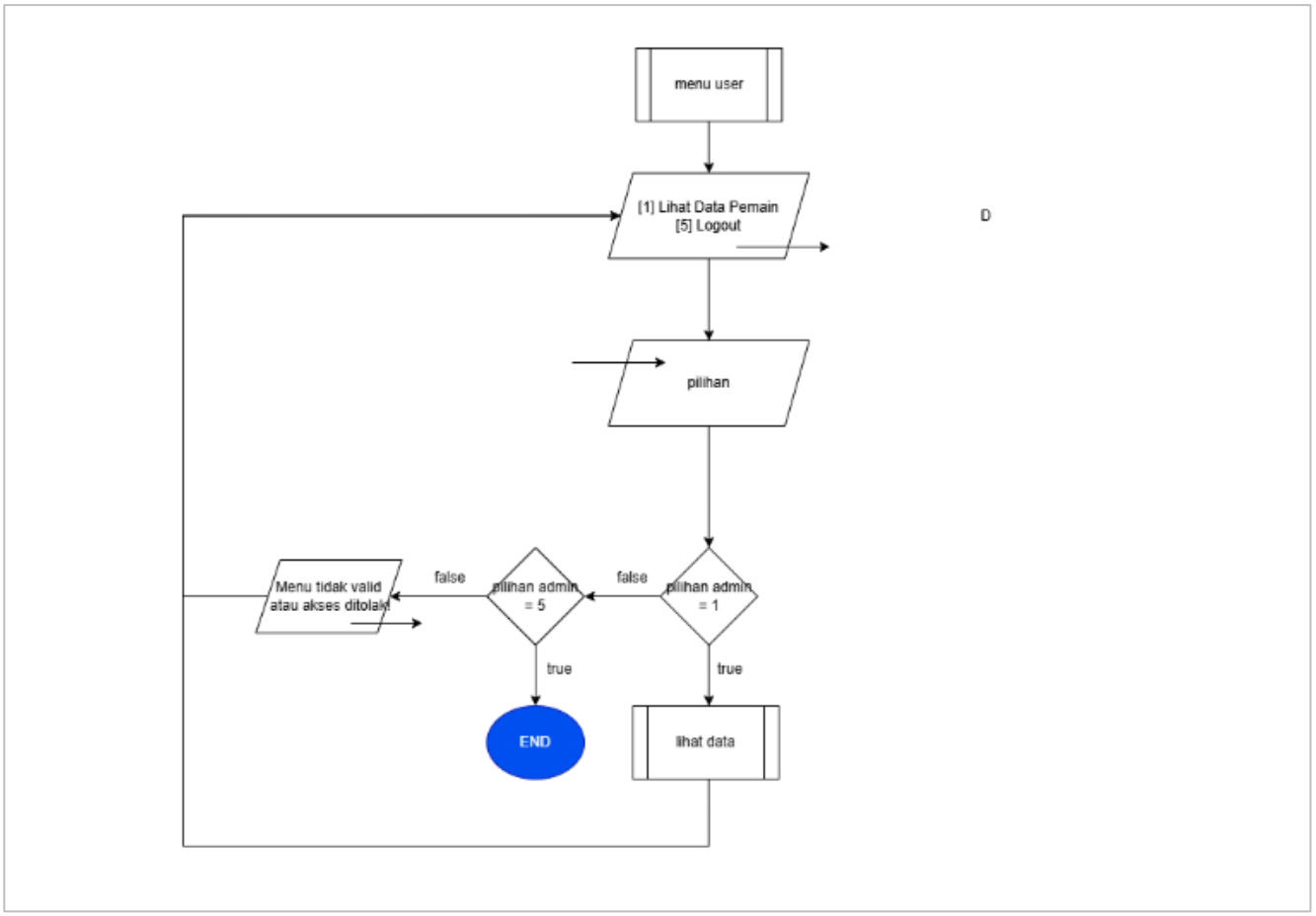
1. Flowchart



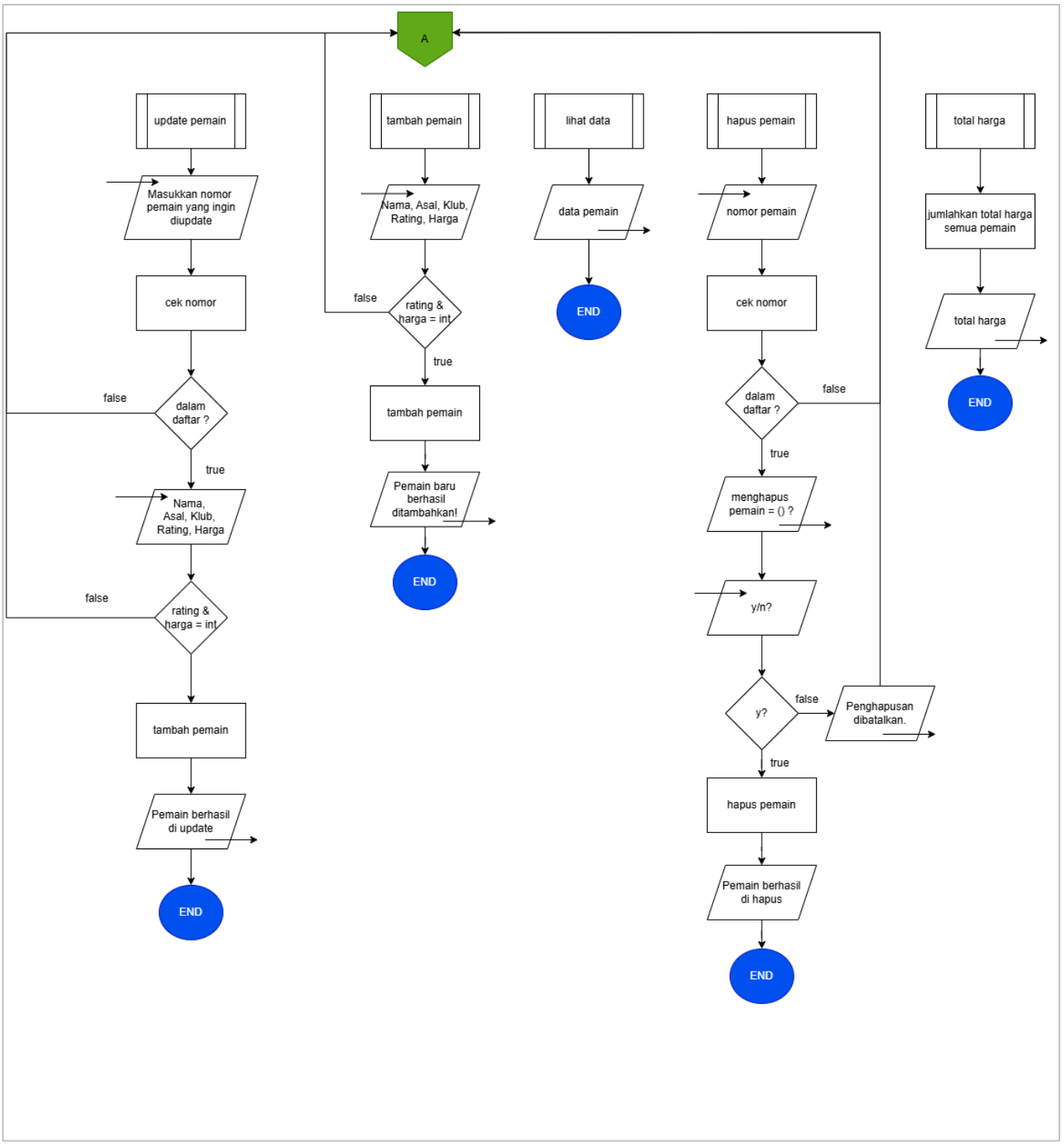
Gambar 1.1 Flowchart Bagian Awal



Gambar 1.2 Flowchart Menu Admin



Gambar 1.3 Flowchart Menu User biasa



Gambar 1.4 Flowchart Fungsi

2. Deskripsi Singkat Program

A. Multiuser (Admin dan User Biasa)

- Admin memiliki hak akses penuh untuk melihat, menambah, mengubah, dan menghapus data pemain.
- User biasa hanya dapat melihat daftar pemain.
- Sistem login membedakan kedua peran ini berdasarkan data akun yang tersimpan.

B. Registrasi Akun Baru

- Pengguna dapat membuat akun baru melalui menu register.
- Akun baru yang terdaftar otomatis berstatus sebagai *user biasa*.

C. Autentikasi Login

- Pengguna harus login terlebih dahulu sebelum masuk ke sistem.
- Jika username atau password salah, sistem akan menolak dan meminta pengguna mencoba lagi.

D. CRUD (Create, Read, Update, Delete)

- ***Create*** → Menambahkan pemain baru dengan data lengkap seperti nama, negara asal, klub sebelumnya, rating, dan harga.
- ***Read*** → Menampilkan seluruh daftar pemain yang tersimpan.
- ***Update*** → Mengubah harga pemain (misalnya saat performa berubah).
- ***Delete*** → Menghapus pemain yang dijual atau dipinjam ke klub lain.

E. Validasi *Input*

- Pengecekan dilakukan dengan perulangan dan kondisi if.

F. Clear Terminal (Tampilan Bersih)

- Menggunakan *import os* dan perintah *os.system('cls')* agar tampilan menu selalu bersih dan rapi setiap kali berpindah halaman.

G. Fungsi *(def), Try, Except,* dan *Error Handling*

- Program menggunakan beberapa fungsi *(def)* untuk memisahkan tugas agar kode lebih rapi, seperti *clear_screen()* untuk membersihkan layar, tampilkan_pemain() untuk menampilkan data pemain, dan total_harga_pemain() untuk menghitung total harga pemain secara rekursif.
- Blok *try* dan *except* digunakan untuk menangani kesalahan input agar program tidak berhenti saat terjadi error. Misalnya saat pengguna memasukkan huruf pada input angka atau saat login salah.

- *Error handling* memberikan pesan kesalahan yang jelas dan membuat program tetap berjalan normal.
- Penggunaan *def* dan *try-except* membuat program lebih terstruktur, aman, dan mudah dikembangkan.

H. Fungsi Rekursif

- Program menggunakan fungsi rekursif bernama *total_harga_pemain()* untuk menghitung total harga seluruh pemain secara otomatis.
- Fungsi ini memanggil dirinya sendiri sampai semua data pemain dijumlahkan.
- Jika data pemain sudah habis (basis rekursi), fungsi akan berhenti dan mengembalikan nilai total.
- Penggunaan rekursi membuat proses perhitungan lebih efisien dan kode menjadi lebih sederhana tanpa perlu perulangan manual.

I. Modularisasi Program (Pemecahan File)

Program dibagi menjadi beberapa file agar struktur lebih rapi dan mudah dikembangkan:

- **main.py** → Program utama yang menjalankan menu dan mengatur alur sistem.
- **login.py** → Berisi fungsi autentikasi login, register, dan tampilan tabel utama menggunakan PrettyTable.
- **data.py** → Menyimpan data awal pemain dan akun.
- **hps.py** → Menyediakan fungsi tambahan seperti clear_screen() dan pengaturan warna

J. Tampilan Berwarna dan Interaktif

Program memanfaatkan dua library utama untuk tampilan terminal yang menarik:

Library	Kegunaan
colorma	Memberi warna pada teks
prettytable	Membuat tampilan tabel seperti daftar pemain dan hasil login agar lebih rapi.
datetime	Menampilkan waktu login secara otomatis

3. Source Code

A. Program Sistem Managemen Tim Bola

```python
pemain = {
    1: {"nama": "Bruno Fernandes", "negara": "Portugal", "klub": "Sporting Lisbon", "rating": 90, "harga": 80000000},
    2: {"nama": "Casemiro", "negara": "Brazil", "klub": "Real Madrid", "rating": 88, "harga": 70000000},
    3: {"nama": "Benjamin Sesko", "negara": "Slovenia", "klub": "Salzburg", "rating": 92, "harga": 85000000},
    4: {"nama": "Kobbie Mainoo", "negara": "Inggris", "klub": "Akademi MU", "rating": 81, "harga": 2500000},
    5: {"nama": "Senne Lammes", "negara": "Belgia", "klub": "Royal Antwerp", "rating": 94, "harga": 18000000},
}
users = {
    "admin": {"password": "024", "role": "admin"},
    "user": {"password": "420", "role": "user"}
}
```

Gambar 3.1
Source Code di file *data,py*

```python
from colorama import Fore, Style, init
from prettytable import PrettyTable
from hps import clear_screen
from data import users
from datetime import datetime

init(autoreset=True, convert=True, strip=False)

def login_user():
    while True:
        clear_screen()
        tabel = PrettyTable()
        tabel.field_names = [Fore.YELLOW + "Menu " + Style.RESET_ALL, Fore.YELLOW + "Ket" + Style.RESET_ALL]
        tabel.add_row(["1", "Login"])
        tabel.add_row(["2", "Registrasi"])
        tabel.add_row(["3", "Keluar"])

        print(Fore.RED + "=== SISTEM LOGIN MANCHESTER UNITED ===" + Style.RESET_ALL)
        print(tabel)
        pilih = input(Fore.CYAN + "Pilih menu: " + Style.RESET_ALL)

        if pilih == "1":
            clear_screen()
            print(Fore.YELLOW + "=== LOGIN ===" + Style.RESET_ALL)
            username = input("Username: ").strip()
            password = input("Password: ").strip()

            if username in users and users[username]["password"] == password:
                waktu = datetime.now().strftime("%d/%m/%Y %H:%M:%S")
                clear_screen()
                tabel_login = PrettyTable()
                tabel_login.field_names = ["Status", "Tanggal & Waktu Login"]
                tabel_login.add_row([" Login Berhasil", waktu])
                print(Fore.GREEN + str(tabel_login))
```

```
                role = users[username]["role"]
                input("\nTekan Enter untuk melanjutkan ke sistem...")
                return username, role
            else:
                input(Fore.RED + " Username atau password salah! Tekan Enter untuk ulang...")

        elif pilih == "2":
            clear_screen()
            print(Fore.YELLOW + "=== REGISTER USER BARU ===" + Style.RESET_ALL)
            new_user = input("Buat username: ").strip()
            if new_user in users:
                input(Fore.RED + " Username sudah digunakan! Tekan Enter...")
                continue

            new_pass = input("Buat password: ").strip()
            role = input("Masukkan role (admin/user): ").lower().strip()
            if role not in ["admin", "user"]:
                input(Fore.RED + " Role tidak valid! Tekan Enter...")
                continue

            users[new_user] = {"password": new_pass, "role": role}
            clear_screen()
            tabel_reg = PrettyTable()
            tabel_reg.field_names = ["Status", "Username", "Role"]
            tabel_reg.add_row([" Registrasi Berhasil", new_user, role])
            print(Fore.GREEN + str(tabel_reg))
            input("\nTekan Enter untuk kembali ke menu login...")

        elif pilih == "3":
            clear_screen()
            tabel_keluar = PrettyTable()
            tabel_keluar.field_names = ["Status", "Pesan"]
            tabel_keluar.add_row([" Keluar", "Terima kasih telah menggunakan sistem MUNYUK!"])
            print(Fore.YELLOW + str(tabel_keluar))
            exit()

        else:
            input(Fore.RED + " Pilihan tidak valid! Tekan Enter untuk ulang...")
```

Gambar 3.2
Source Code di file *login,py*

```
from prettytable import PrettyTable
import random

def tampilkan_pemain(data_pemain):
    tabel = PrettyTable()
    tabel.field_names = ["No", "Nama", "Negara", "Klub Sebelumnya", "Rating", "Harga (€)"]
    for i, data in data_pemain.items():
        tabel.add_row([i, data['nama'], data['negara'], data['klub'], data['rating'], data['harga']])
    print(tabel)

def tambah_pemain(pemain):
    print("=== TAMBAH PEMAIN BARU ===")
    nama = input("Nama pemain: ")
    negara = input("Asal negara: ")
    klub = input("Klub sebelumnya: ")

    try:
        rating_input = input("Rating (kosongkan untuk acak): ")
        rating = int(rating_input) if rating_input else random.randint(70, 99)
        harga_input = input("Harga (€) (kosongkan untuk acak): ")
        harga = int(harga_input) if harga_input else random.randint(1_000_000, 100_000_000)
```

```
        id_baru = max(pemain.keys()) + 1 if pemain else 1
        pemain[id_baru] = {
            "nama": nama, "negara": negara, "klub": klub,
            "rating": rating, "harga": harga
        }
        input("Pemain baru berhasil ditambahkan!")
    except ValueError:
        input("Rating dan harga harus berupa angka!")

def hapus_pemain(pemain):
    print("=== HAPUS PEMAIN ===")
    try:
        nomor = int(input("Masukkan nomor pemain yang akan dihapus: "))
        if nomor in pemain:
            print(f"Menghapus pemain: {pemain[nomor]['nama']}")
            yakin = input("Yakin (y/n)? ").lower()
            if yakin == "y":
                del pemain[nomor]
                input("Pemain berhasil dihapus!")
            else:
                input("Dibatalkan.")
        else:
            input("Nomor pemain tidak valid!")
    except ValueError:
        input("Masukkan angka yang benar!")

def update_pemain(pemain):
    print("=== UPDATE DATA PEMAIN ===")
    try:
        nomor = int(input("Masukkan nomor pemain yang ingin diupdate: "))
        if nomor in pemain:
            data = pemain[nomor]
            print(f"\nData pemain saat ini: {data}")
            print("Kosongkan input jika tidak ingin mengubah field tersebut.\n")

            nama_baru = input(f"Nama baru [{data['nama']}]: ") or data['nama']
            negara_baru = input(f"Negara baru [{data['negara']}]: ") or data['negara']
            klub_baru = input(f"Klub baru [{data['klub']}]: ") or data['klub']
            rating_baru = input(f"Rating baru [{data['rating']}]: ") or str(data['rating'])
            harga_baru = input(f"Harga baru (€) [{data['harga']}]: ") or str(data['harga'])

            if rating_baru.isdigit() and harga_baru.isdigit():
                pemain[nomor] = {
                    "nama": nama_baru,
                    "negara": negara_baru,
                    "klub": klub_baru,
                    "rating": int(rating_baru),
                    "harga": int(harga_baru)
                }
                input("Data pemain berhasil diperbarui!")
            else:
                input("Rating dan harga harus berupa angka!")
        else:
            input("Nomor pemain tidak ditemukan!")
    except ValueError:
        input("Masukkan angka yang benar!")
```

Gambar 3.3
Source Code di file *pemain,py*

```python
import os
from prettytable import PrettyTable
from colorama import Fore, Style, init

init(autoreset=True, convert=True, strip=False)

def total_harga_pemain(pemain):
    return sum(data["harga"] for data in pemain.values())

def tampilkan_pemain(pemain):
    clear_screen()
    print(Fore.RED + "=== DAFTAR PEMAIN MANCHESTER UNITED ===" + Style.RESET_ALL)
    tabel = PrettyTable()
    tabel.field_names = ["No", "Nama", "Negara", "Klub", "Rating", "Harga (€)"]
    for i, data in pemain.items():
        tabel.add_row([
            i,
            data["nama"],
            data["negara"],
            data["klub"],
            data["rating"],
            f"€{data['harga']:,}"
        ])
    print(Fore.CYAN + str(tabel) + Style.RESET_ALL)

def tampilkan_user(users):
    print(Fore.YELLOW + "\n=== DAFTAR USER TERDAFTAR ===" + Style.RESET_ALL)
    tabel = PrettyTable()
    tabel.field_names = ["Username", "Role"]
    for username, data in users.items():
        tabel.add_row([username, data["role"]])
    print(Fore.CYAN + str(tabel) + Style.RESET_ALL)

def clear_screen():
    os.system("cls" )
```

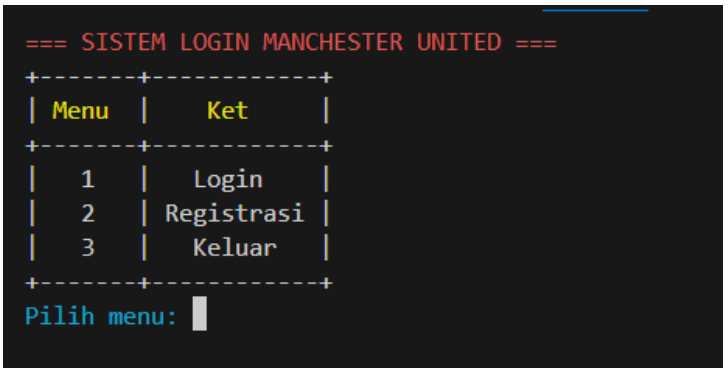
Gambar 3.4
Source Code di file *hpss,py*

```python
from colorama import Fore, Style, init
from login import login_user, clear_screen
from hps import tampilkan_pemain, total_harga_pemain
from data import pemain

init(autoreset=True, convert=True, strip=False)

user_login, role = login_user()

while True:
    clear_screen()
    print(Fore.RED + "=== MANCHESTER UNITED MANAGEMENT SYSTEM ===")
    print(Fore.GREEN + f"Login sebagai: {user_login} ({role})")
    print("-----------------------------------------------")
    print("[1] Lihat Data Pemain")
    if role == "admin":
        print("[2] Tambah Pemain Baru")
        print("[3] Update Data Pemain")
        print("[4] Hapus Pemain")
        print("[5] Lihat Total Harga Semua Pemain")
    print("[6] Logout")
    print("-----------------------------------------------")

    menu = input(Fore.BLUE + "Pilih menu: " + Style.RESET_ALL)

    if menu == "1":
        tampilkan_pemain(pemain)
        input("\nTekan Enter untuk kembali...")

    elif menu == "2" and role == "admin":
        clear_screen()
        print(Fore.YELLOW + "=== TAMBAH PEMAIN BARU ===")
        nama = input("Nama pemain: ")
        negara = input("Asal negara: ")
        klub = input("Klub sebelumnya: ")
        rating = int(input("Rating (0-100): "))
        harga = int(input("Harga (€): "))
        id_baru = max(pemain.keys()) + 1
        pemain[id_baru] = {"nama": nama, "negara": negara, "klub": klub, "rating": rating, "harga": harga}
        input(Fore.GREEN + " Pemain berhasil ditambahkan!")

    elif menu == "3" and role == "admin":
        tampilkan_pemain(pemain)
        nomor = int(input("Masukkan nomor pemain: "))
        if nomor in pemain:
            data = pemain[nomor]
            print("\nKosongkan jika tidak ingin mengubah field.")
            nama_baru = input(f"Nama [{data['nama']}]: ") or data["nama"]
            negara_baru = input(f"Negara [{data['negara']}]: ") or data["negara"]
            klub_baru = input(f"Klub [{data['klub']}]: ") or data["klub"]
            rating_baru = input(f"Rating [{data['rating']}]: ") or data["rating"]
            harga_baru = input(f"Harga [{data['harga']}]: ") or data["harga"]
            pemain[nomor] = {
                "nama": nama_baru, "negara": negara_baru, "klub": klub_baru,
                "rating": int(rating_baru), "harga": int(harga_baru)
            }
            input(Fore.GREEN + " Data pemain diperbarui!")

    elif menu == "4" and role == "admin":
        tampilkan_pemain(pemain)
        nomor = int(input("Masukkan nomor pemain yang ingin dihapus: "))
        if nomor in pemain:
            del pemain[nomor]
            input(Fore.GREEN + " Pemain berhasil dihapus!")
```

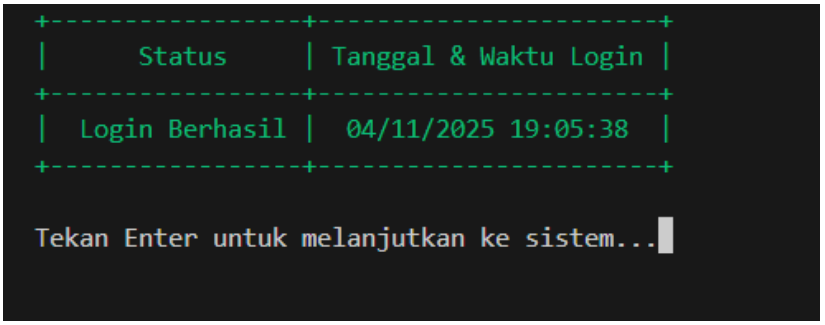
```
65
66      elif menu == "5" and role == "admin":
67          clear_screen()
68          total = total_harga_pemain(pemain)
69          print(Fore.MAGENTA + f" Total harga semua pemain: €{total:,}")
70          input("\nTekan Enter untuk kembali...")
71
72      elif menu == "6":
73          clear_screen()
74          print(Fore.YELLOW + "Logout berhasil. Terima kasih telah menggunakan sistem MUNYUK!")
75          break
76
77      else:
78          input(Fore.RED + " Menu tidak valid atau akses ditolak!")
```

Gambar 3.5
Source Code di file utama : *main,py*

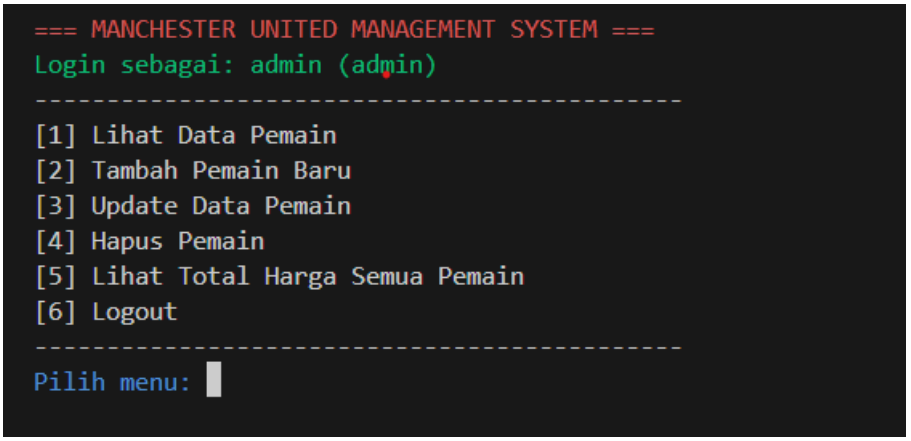
4. Hasil Output



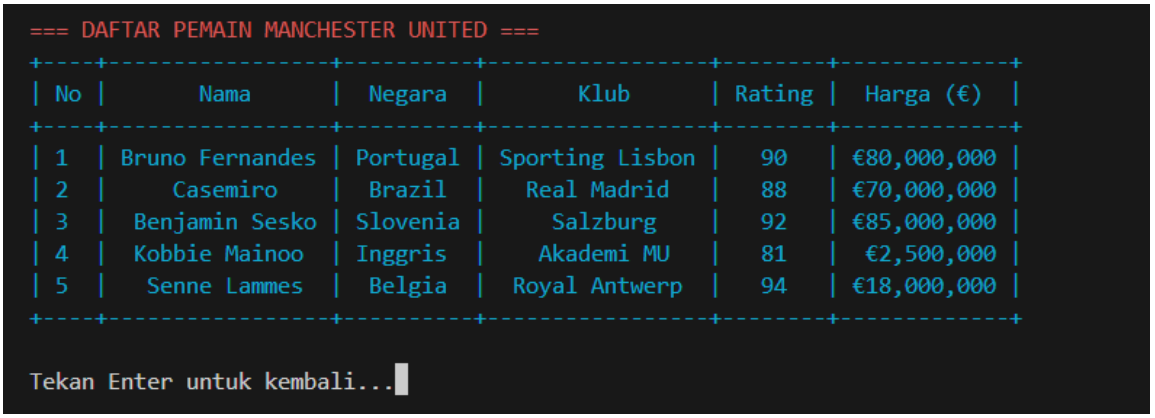
Gambar 4.1
Tampilan Awal setelah program dimulai



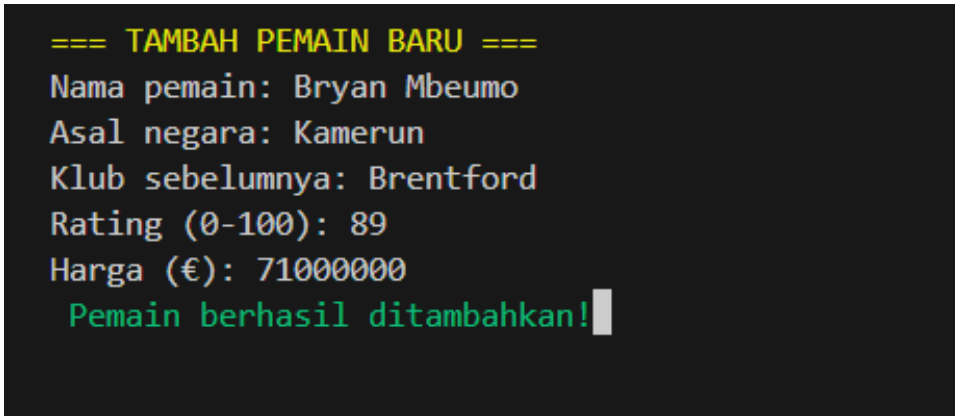
Gambar 4.2
Tampilan Login Berhasil



Gambar 4.3
Hasil *Login* berhasil sebagai admin

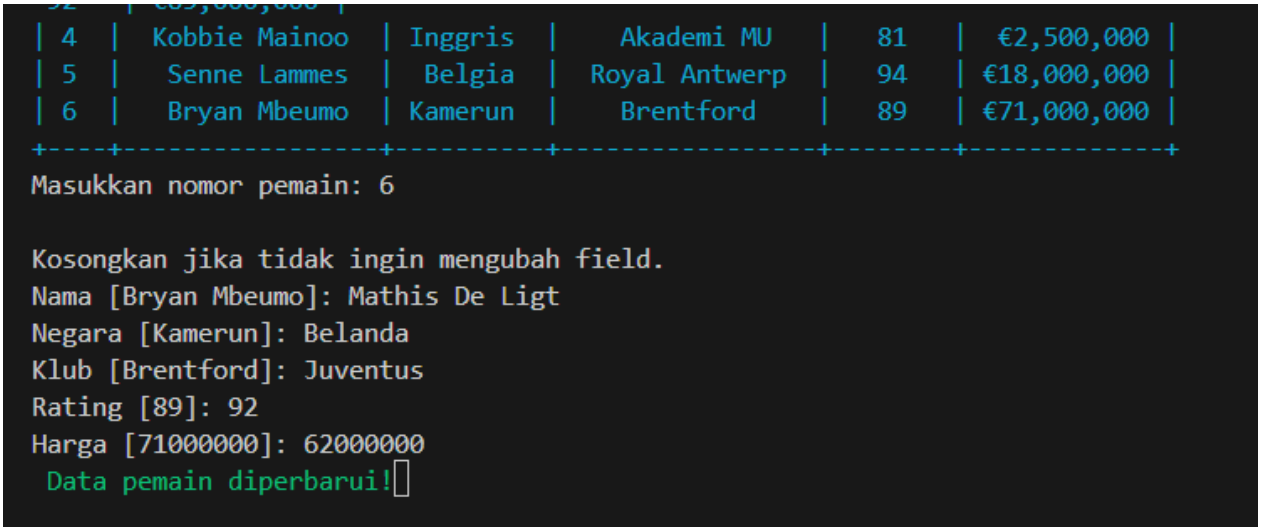


Gambar 4.4
Tampilan Lihat Data Pemain *(Read)*



Gambar 4.5
Tampilan Menambah Pemain Baru *(Create)*

Gambar 4.6
Tampilan data setelah di tambahkan dan sebelum di *update*

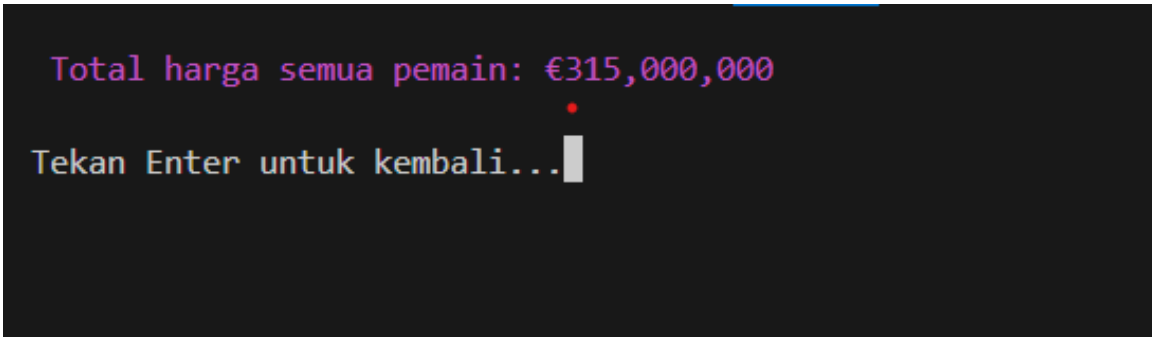


Gambar 4.7
Tampilan ingin *(Update)* Data Pemain

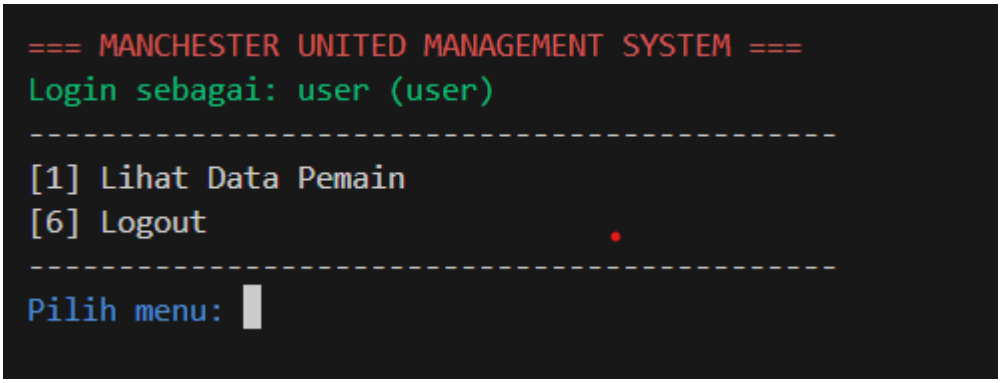
Gambar 4.8
Tampilan seluruh data Setelah di *update* dan sebelum di jual *(Delete)*

Gambar 4.9
Tampilan Pemain yang ingin dijual/dipinjam *(Delete)*

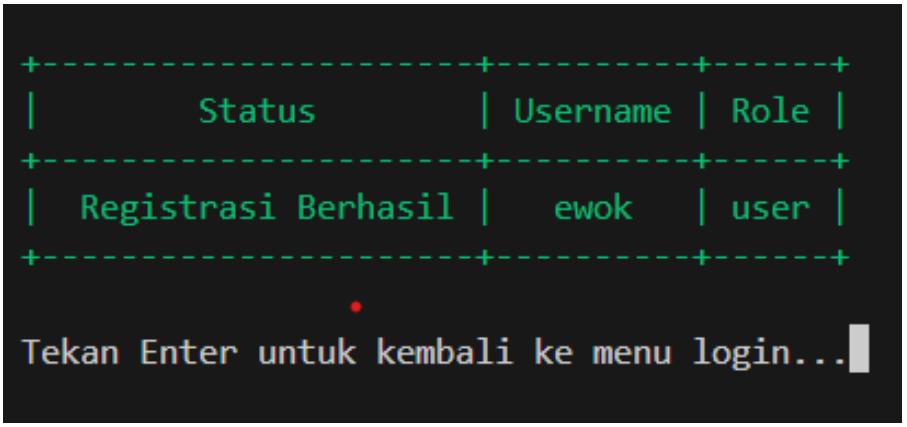
Gambar 4.10
Tampilan Data Pemain yang sudah di jual *(Delete)*



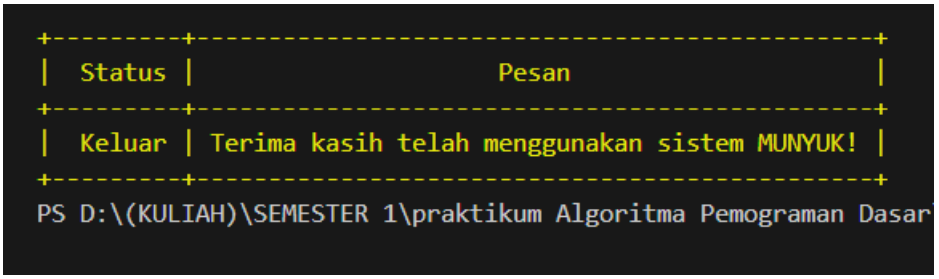
Gambar 4.11
Tampilan Total Semua Harga Pemain (Rekursif)



Gambar 4.12
Tampilan *Login* sebagai user (hanya bisa lihat data pemain dan *logout*)



Gambar 4.13
Tampilan Registrasi atau buat akun baru



Gambar 4.14
Tampilan setelah Keluar dari program

5. Langkah-langkah GIT

5.1 GIT Init

```
Asus@DESKTOP-8GBFIO2 MINGW64 /d/(KULIAH)/SEMESTER 1/praktikum Algoritma Pemogram
an Dasar (main)
$ git init
Reinitialized existing Git repository in D:/(KULIAH)/SEMESTER 1/praktikum Algori
tma Pemograman Dasar/.git/
```

Git init agar bisa *track progress* yag ada pada folder tersebut

5.2 GIT Add

```
Asus@DESKTOP-8GBFIO2 MINGW64 /d/(KULIAH)/SEMESTER 1/praktikum Algoritma Pemogram
an Dasar (main)
$ git add .
```

Git Add menambah semua file yang ada didalam folder sementara sebelum file dikomit dalam repository

5.3 GIT Commit

```
Asus@Nabonkey MINGW64 /d/(KULIAH)/SEMESTER 1/praktikum Algoritma Pemograman Dasa
r (main)
$ git commit -m "update"
[main f8130e3] update
 5 files changed, 369 insertions(+)
 create mode 100644 praktikum-apd/kelas/pertemuan 7/coba 1.py
 create mode 100644 praktikum-apd/kelas/pertemuan 7/main.py
 create mode 100644 praktikum-apd/kelas/pertemuan 7/studi kasus.py
 create mode 100644 praktikum-apd/post-test/post-test-apd-7/2509106024-FERLYAHMA
DNABIL-APD-7.pdf
 create mode 100644 praktikum-apd/post-test/post-test-apd-7/2509106024-FERLYAHMA
DNABIL-APD-7.py
```

Git commit untuk menyimpan perubahan dalam *repository* dengan commit dan pesan *"update"*.

5.4 GIT Remote

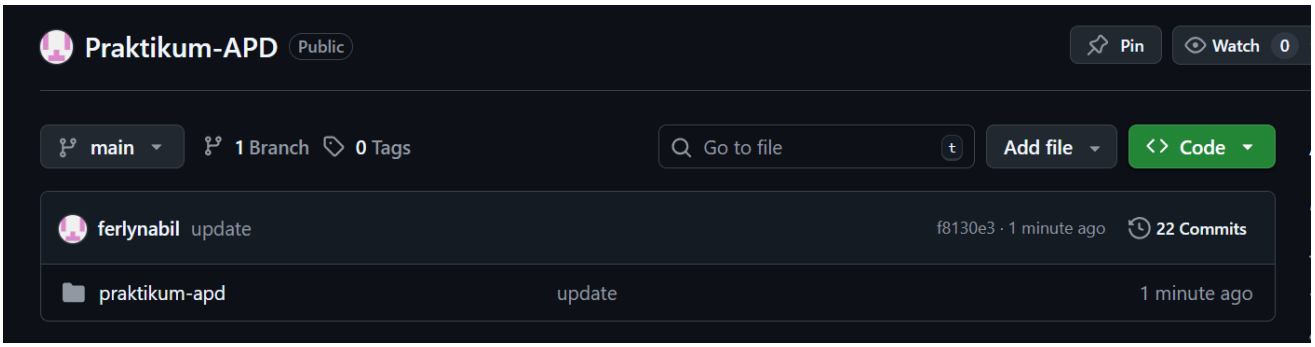
```
Asus@DESKTOP-8GBFIO2 MINGW64 /d/(KULIAH)/SEMESTER 1/praktikum Algoritma Pemogram
an Dasar (main)
$ git remote add origin https://github/ferlynabil/Praktikum-APD.git
```

Git Remote digunakan untuk menambahkan *repository remote* dengan nama *origin.*

5.5 GIT Push

```
Asus@Nabonkey MINGW64 /d/(KULIAH)/SEMESTER 1/praktikum Algoritma Pemograman Dasa
r (main)
$ git push origin main
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 16 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (12/12), 734.42 KiB | 28.25 MiB/s, done.
Total 12 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To https://github.com/ferlynabil/Praktikum-APD.git
   0537909..f8130e3  main -> main
```

GIT Push untuk mengunggah (*push)* perubahan ke *repository* remote pada *branch main.*



Semua Folder sudah ter *upload* pada Github