LAPORAN PRAKTIKUM

POSTTEST 7

ALGORITMA PEMROGRAMAN DASAR

Disusun oleh:

Yoga Ananda Prasetya (2509106017)

Kelas (A1‘25)

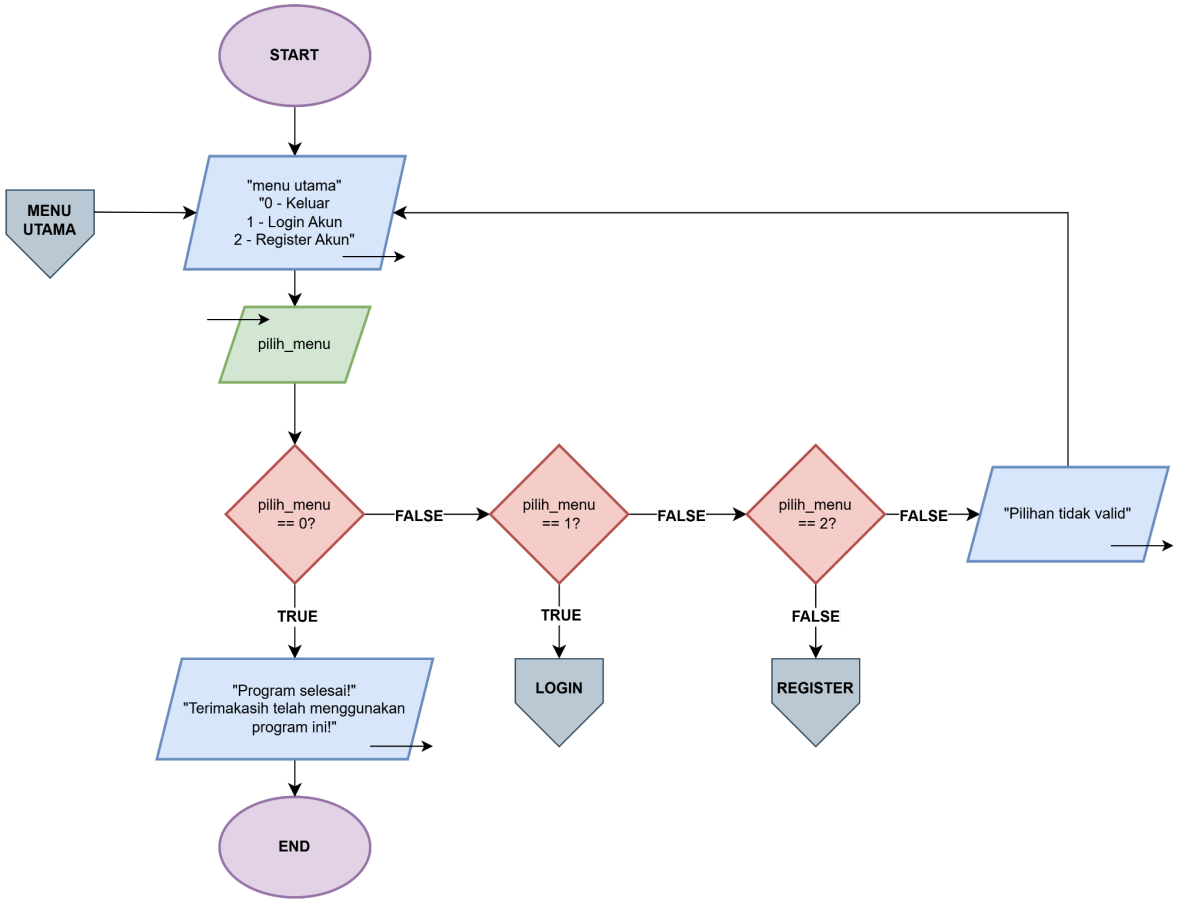
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

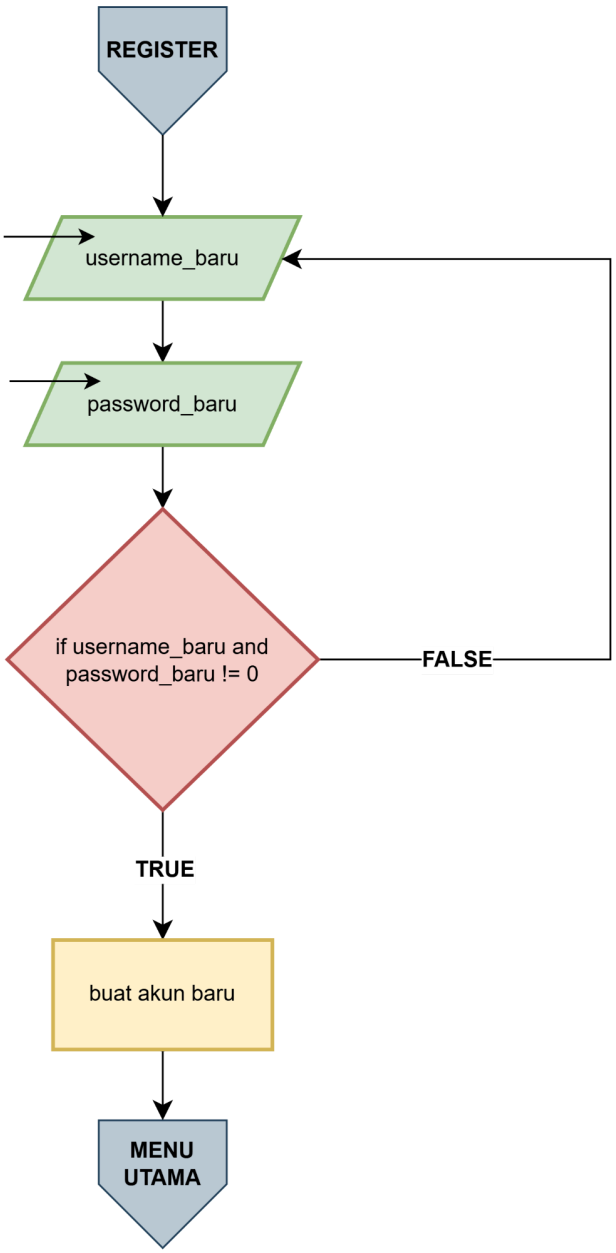
SAMARINDA

2025

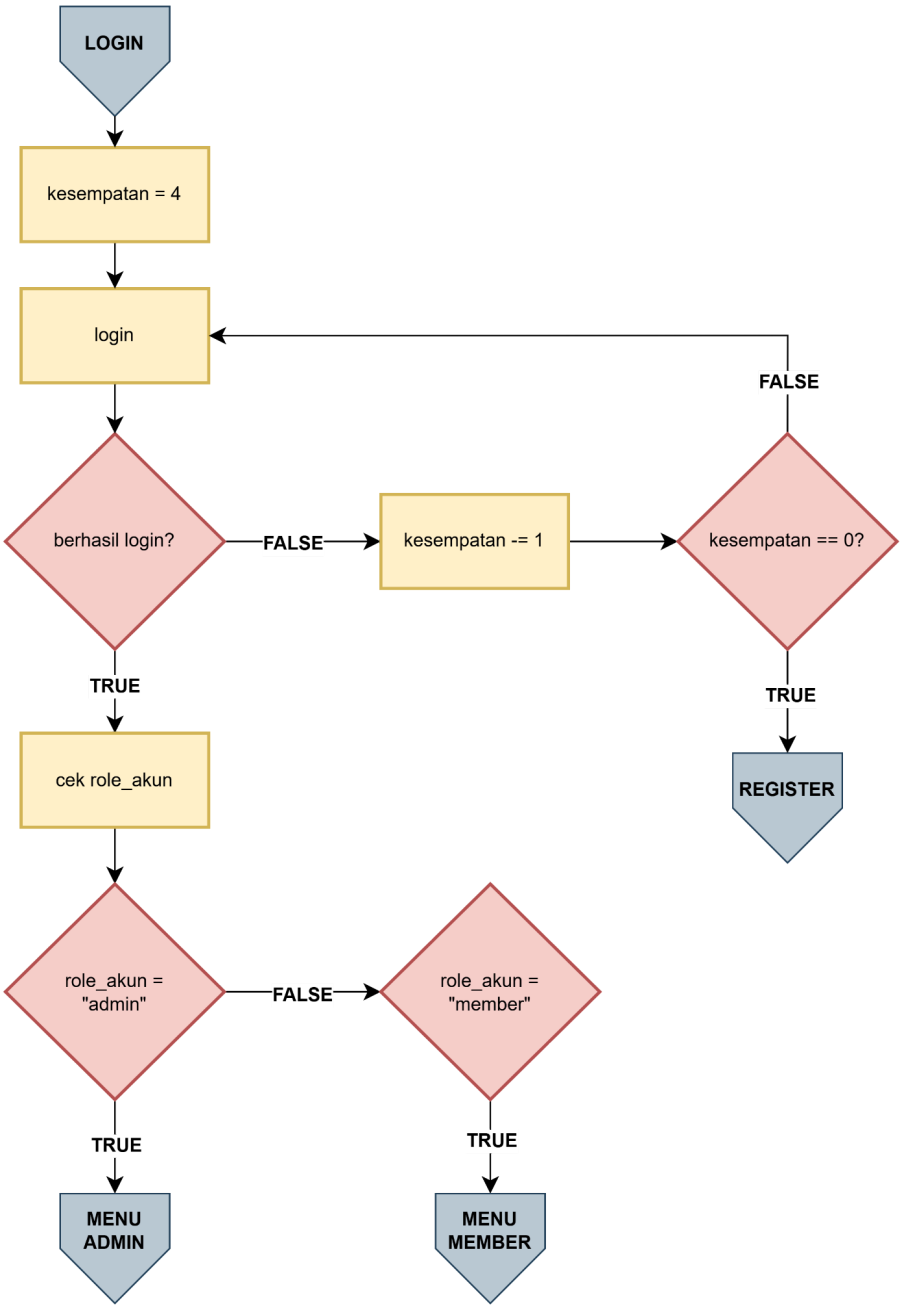
1. Flowchart



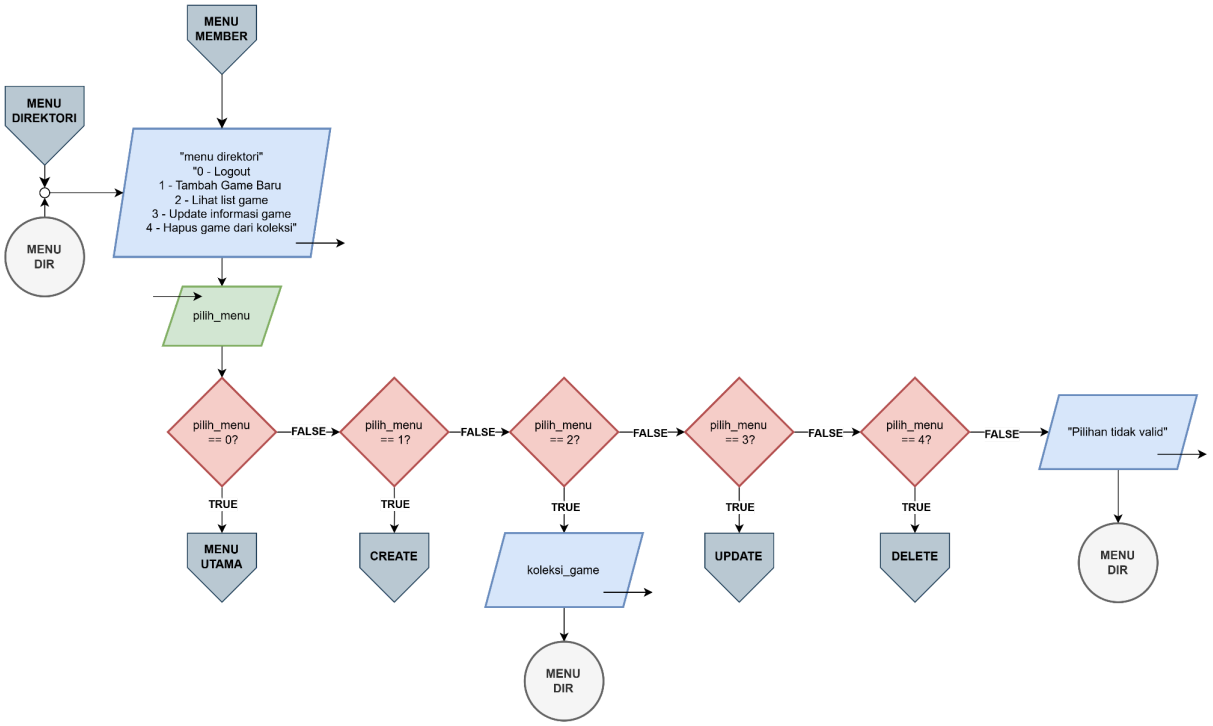
Gambar 1 Flowchart – Menu Awal



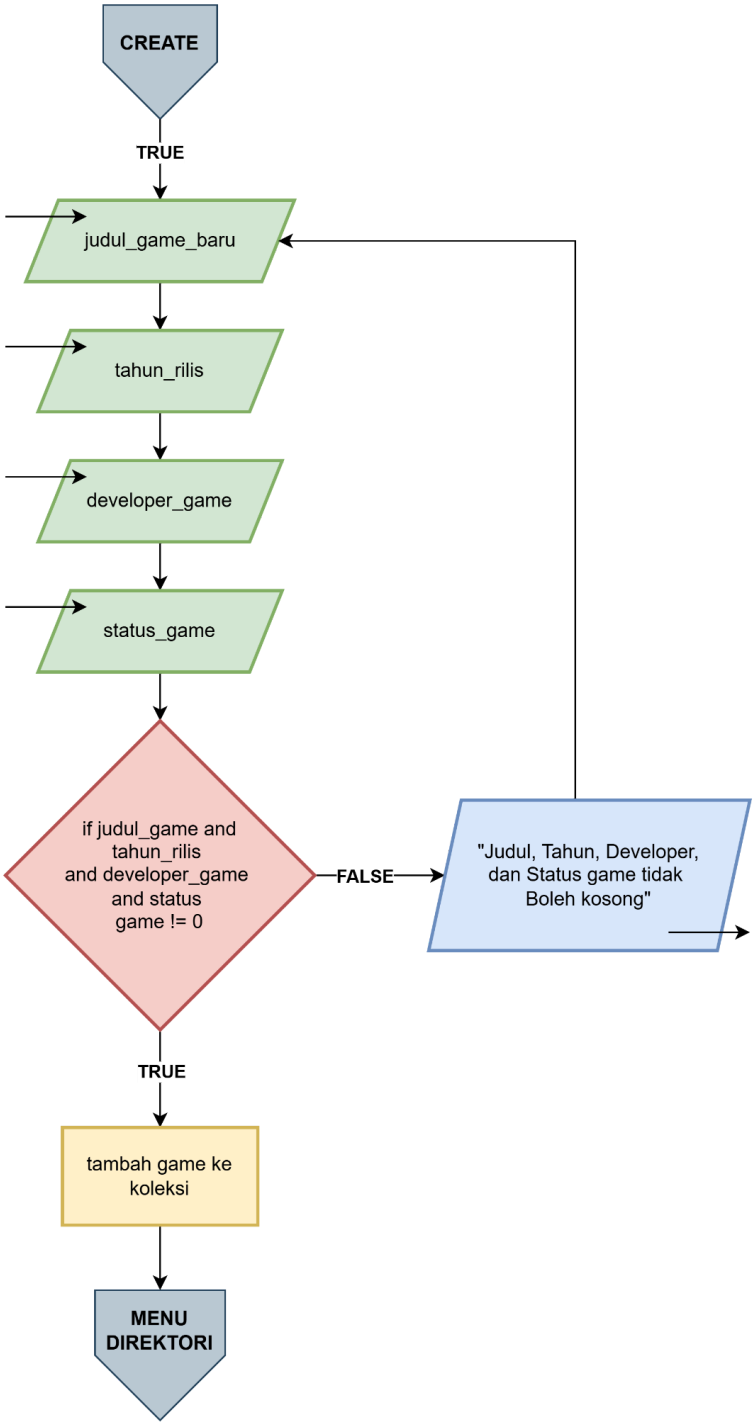
Gambar 2 Flowchart – Menu Register



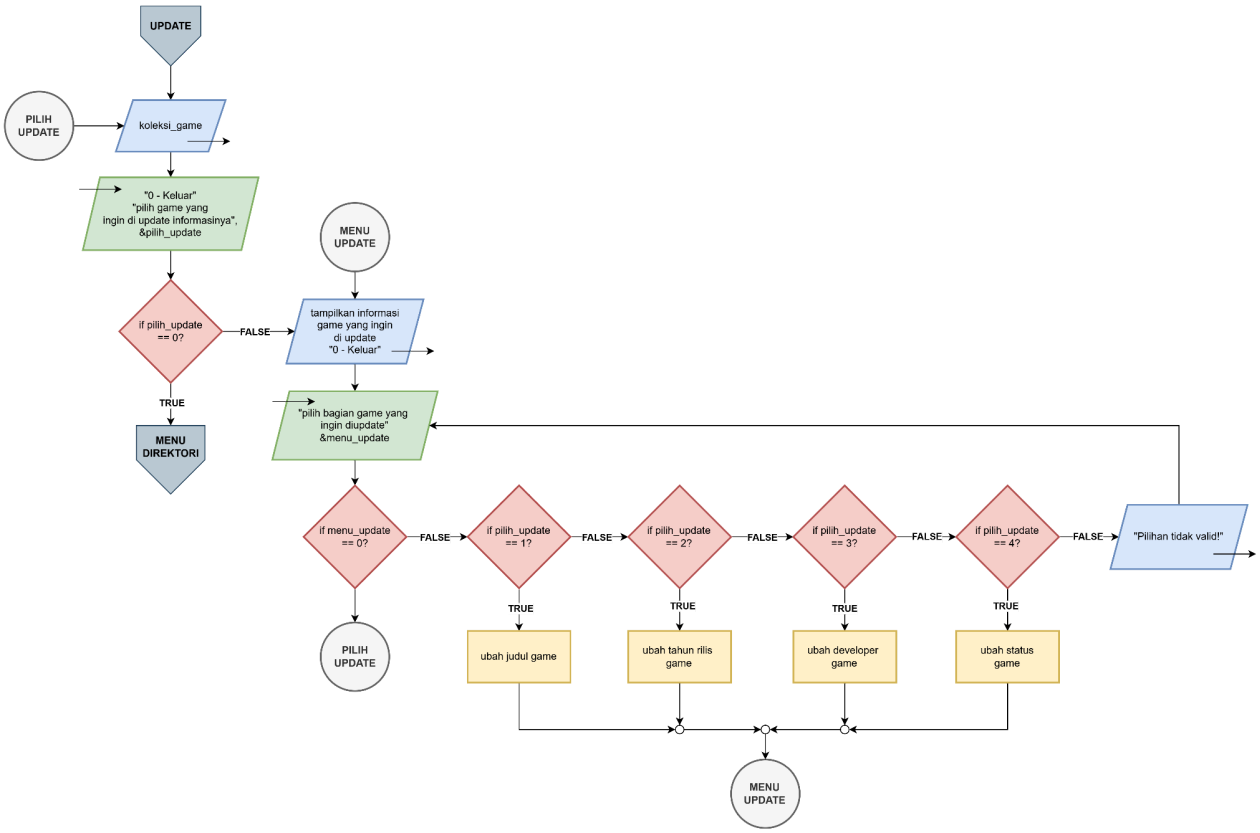
Gambar 3 Flowchart – Menu Login



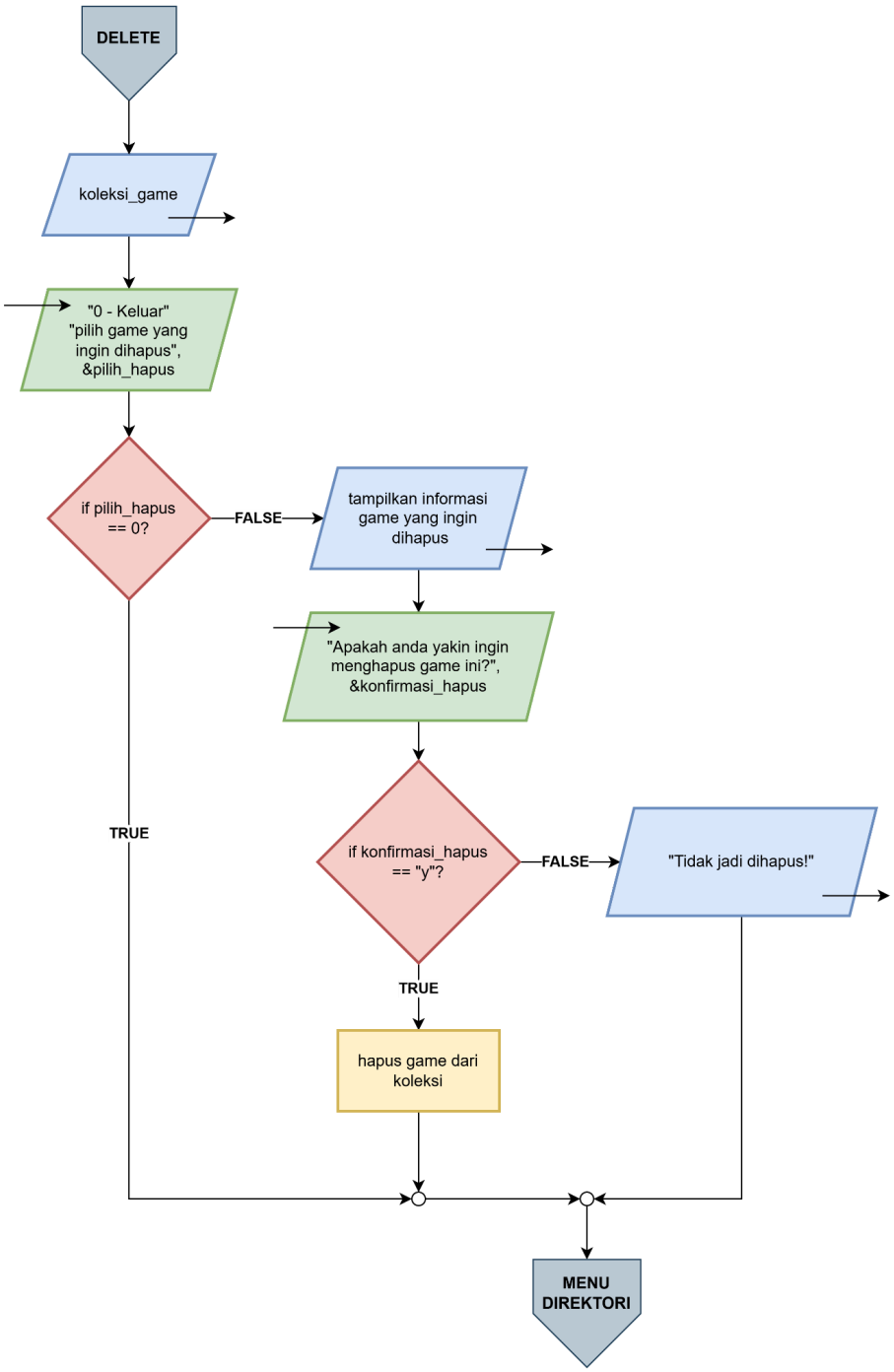
Gambar 4 Flowchart – Menu jika pengguna member



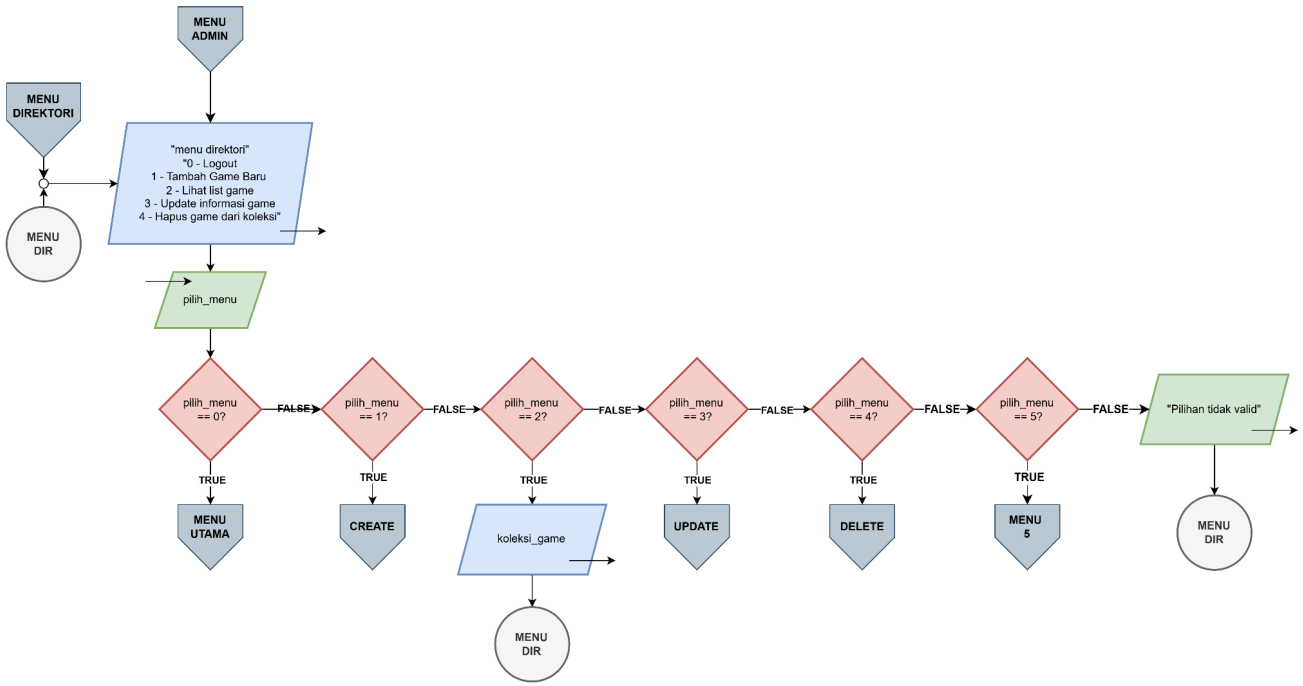
Gambar 5 Flowchart – Menambah game (Create)



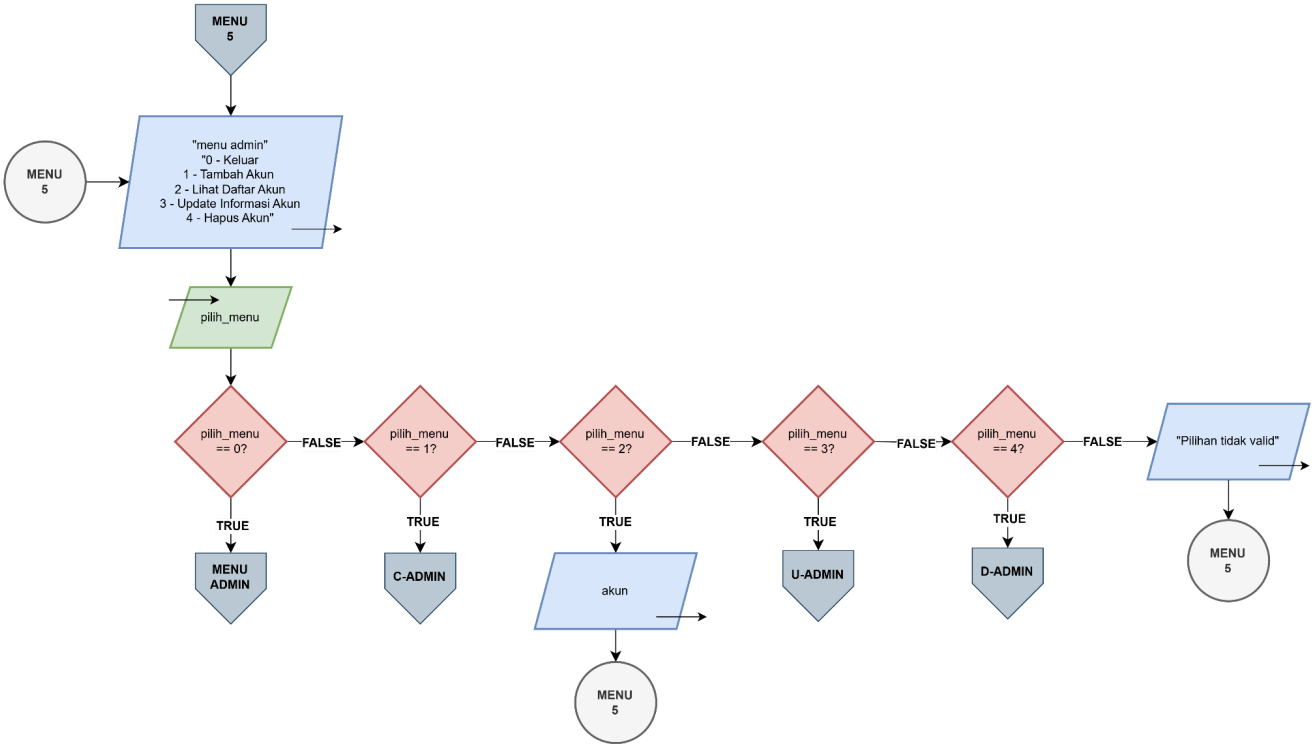
Gambar 6 Flowchart – Mengubah informasi game (Update)



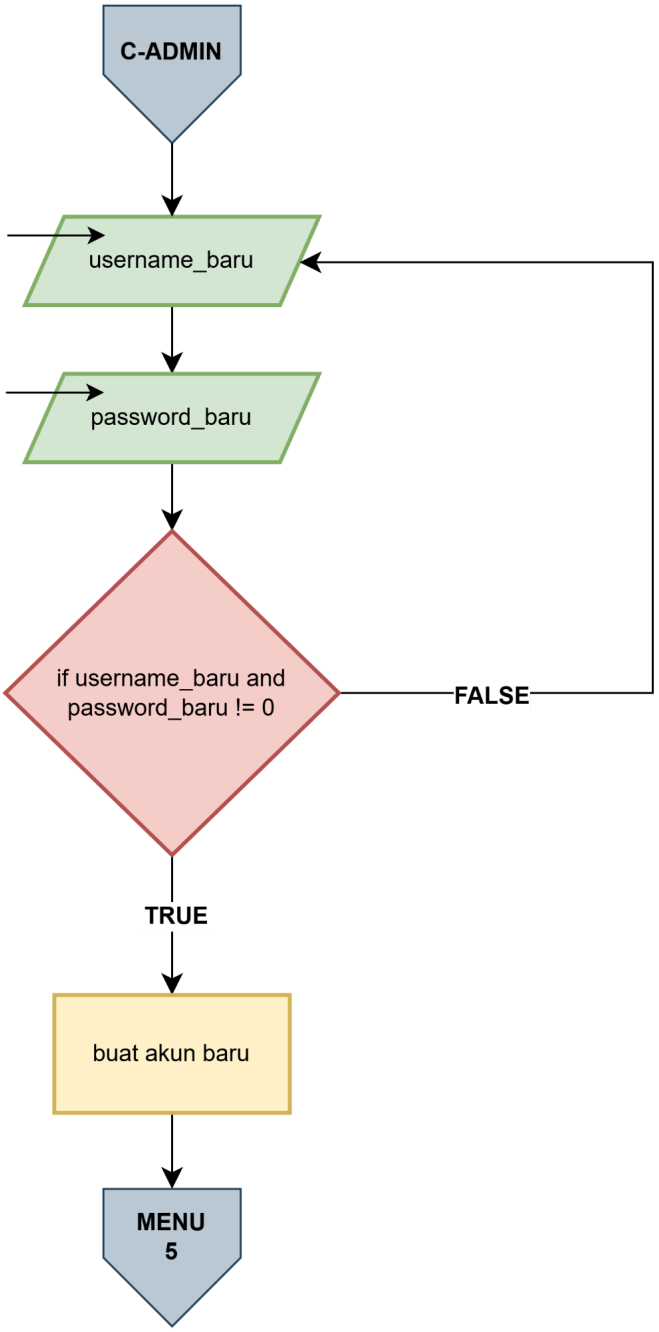
Gambar 7 – Menghapus game (Delete)



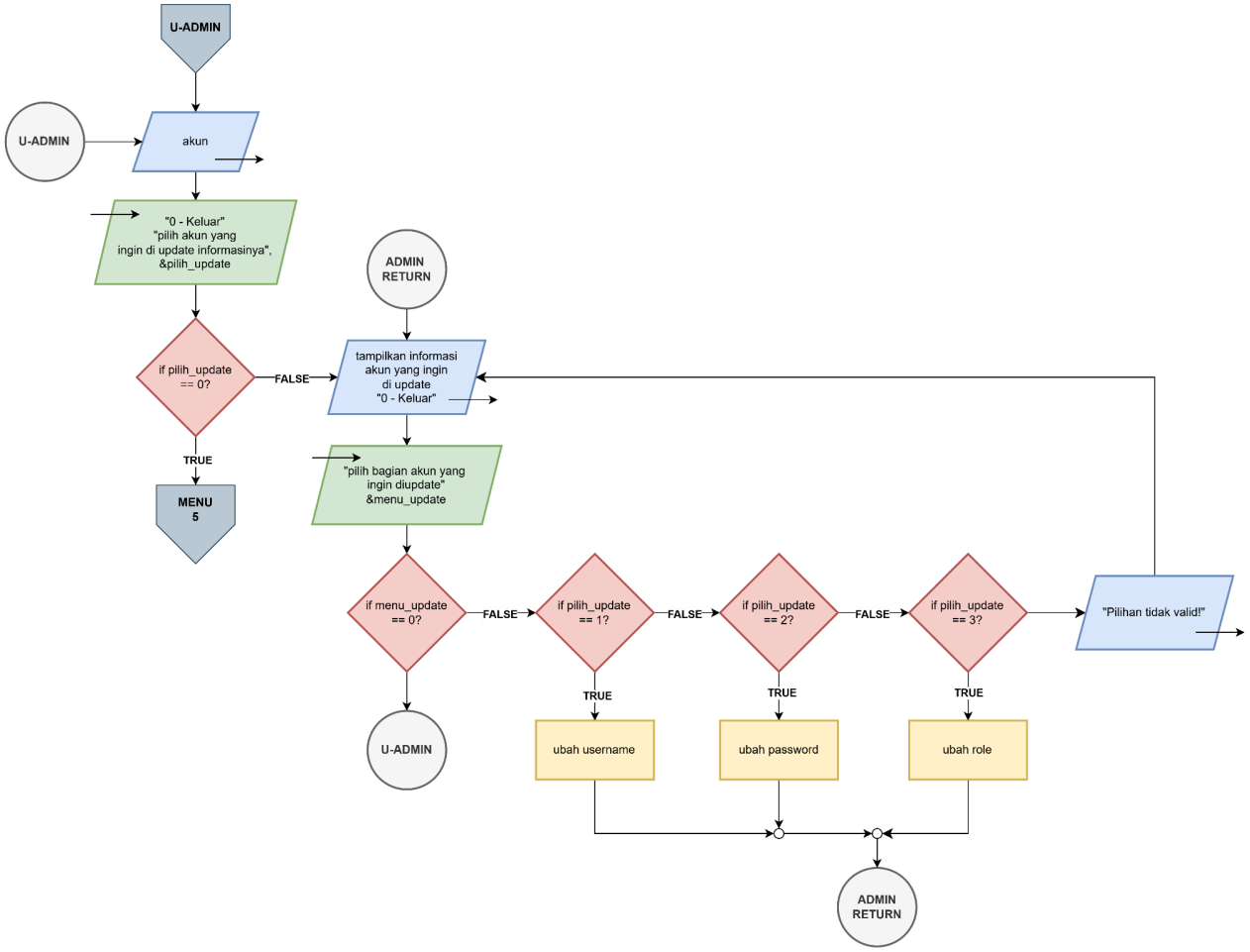
Gambar 8 – Tampilan menu jika pengguna adalah admin



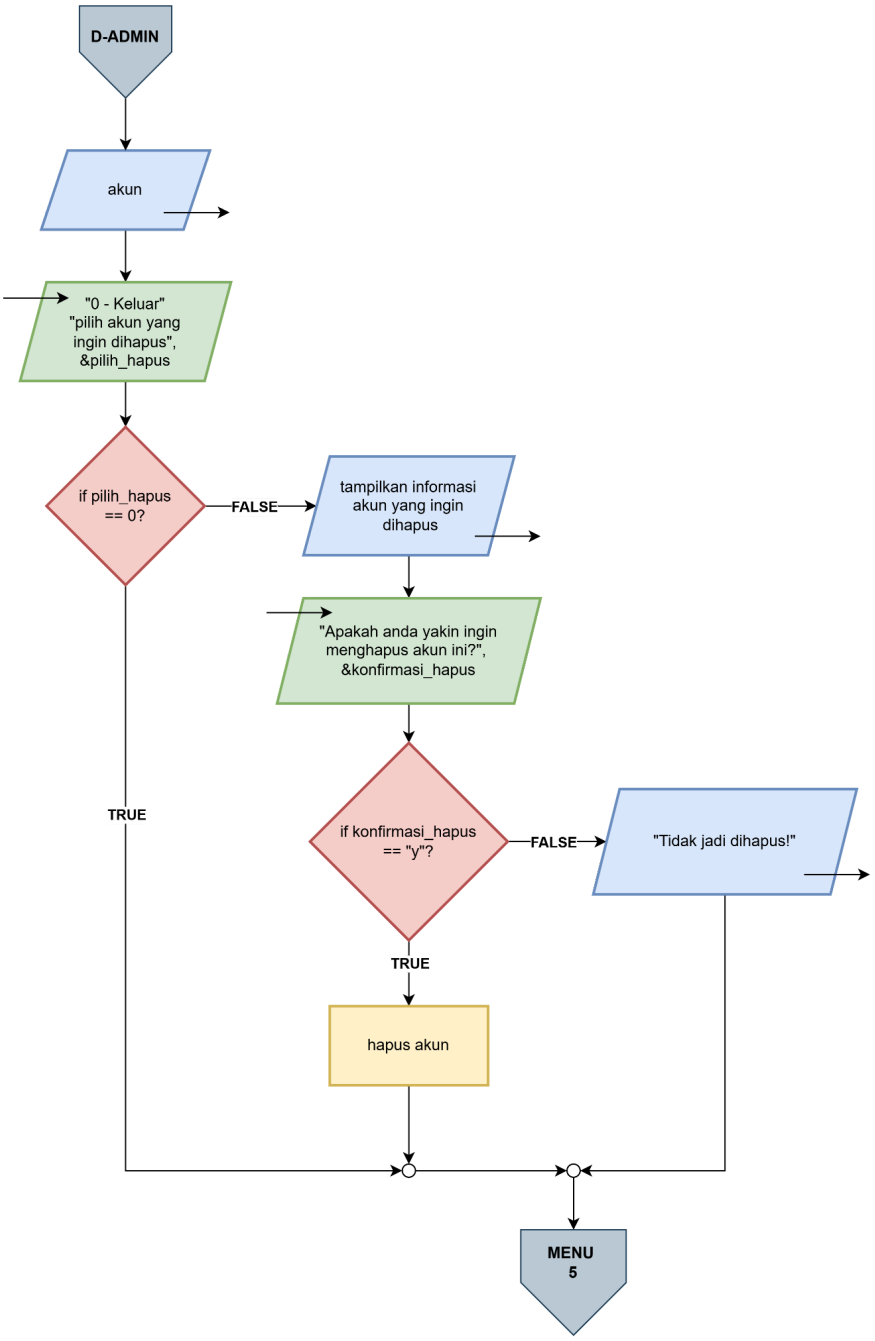
Gambar 9 – Menu Khusus admin



Gambar 10 – Menambah akun (Create)



Gambar 11 – Mengubah informasi akun (Update)



Gambar 12 – Menghapus akun (Delete

2. Deskripsi Singkat Program

Program ini merupakan program yang menerapkan prinsip CRUD (*Create*, *Read*, *Update*, dan *Delete*), yang dimana prinsip CRUD ini diterapkan untuk membuat sebuah sistem direktori sederhana yang dapat menyimpan informasi game yang dimiliki oleh seseorang. Berikut ini merupakan implementasi dari setiap prinsip CRUD pada sistem direktori penyimpanan game.

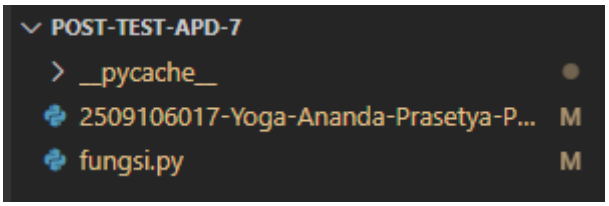
PRINSIP CRUD		IMPLEMENTASI PRINSIP CRUD
Create	:	Pengguna yang terdaftar sebagai member dapat menambahkan game yang diinginkan ke dalam direktori. Disini ada 4 data yang harus dimasukkan pengguna ketika ingin menambah game baru ke direktori. 4 Data tersebut yaitu sebagai berikut : 1. Nama game 2. Tahun rilis game 3. Pengembang/*Developer* game 4. Status game (sudah tamat atau belum)
Read	:	Pengguna dapat melihat daftar game yang sudah ditambahkan ke direktori. Pengguna dapat melihat keempat data yang telah dimasukkan pada proses *create*.
Update	:	Pengguna dapat mengganti data game yang ada di direktori jika seandainya pengguna salah memasukkannya ketika sedang dalam proses *create*. Program juga dilengkapi fitur memilih bagian data mana yang akan diganti, sehingga pengguna tidak perlu repot memasukkan keempat data tersebut jikalau hanya satu data saja yang bermasalah.
Delete	:	Pengguna dapat menghapus game yang terdapat di dalam direktori. Proses *Delete* berlaku secara permanen, artinya data game akan hilang selamanya jika pengguna memilih untuk menghapusnya. Namun, program dilengkapi mekanisme pengaman yang memastikan kembali apakah pengguna ingin menghapus game yang dipilih.

Dalam program terdapat pembagian *role* atau wewenang, yaitu *admin* dan *member*. Untuk *member* sendiri hanya memiliki hak untuk menambah (Create), melihat (Read), mengubah (Update), dan menghapus (Delete) game yang ada di direktori. Sedangkan *admin* memiliki wewenang serta hak yang lebih banyak, seperti membuat akun baru (diluar dari register akun), melihat data akun yang aktif, mengubah informasi akun (seperti password,

username atau bahkan role), serta menghapus akun. Jadi *admin* memiliki dua implementasi CRUD. Implementasi pertama adalah untuk menambah, melihat, mengubah dan menghapus koleksi game yang ada di direktorinya, sedangkan implementasi kedua adalah untuk menambah, melihat, mengubah serta menghapus akun yang terdaftar pada direktori.

3. Source Code

Karena sekarang menggunakan fungsi, maka kode program dipisah menjadi dua file, yaitu file utama (*2509106017-Yoga-Ananda-Prasetya-PT-7.py*) yang berperan sebagai kode utama. File kedua merupakan yang menyimpan deretan fungsi yang akan digunakan di file utama (*fungsi.py*).



Struktur file setelah kode dipisah

3.1. Fitur Utama

3.1.1. Import Library dan Deklarasi Dictionary

Program ini diawali dengan *import* library serta deklarasi dictionary yang akan digunakan. Library yang digunakan adalah *os* dan *time*, keduanya berperan pada proses membersihkan terminal agar hasil *output* terlihat bersih dan rapi. Hanya ada satu dictionary yang digunakan untuk program ini, yaitu dictionary yang digunakan untuk menyimpan data-data akun pengguna. *import fungsi* berfungsi untuk memanggil fungsi-fungsi yang ada pada file *fungsi.py*.

Source Code:

```
# Import library yang diperlukan
import os
import time
import fungsi
from fungsi import *

# Deklarasi Dictionary yang diperlukan
```

```
akun = {
    "acc1": {
        "username": "Chryse",
        "password": "79-Au",
        "role": "admin",
        "game": {
            "g1": {"nama": "Armored Core Verdict Day", "tahun": "2007", "dev":
"Fromsoftware", "status": "Tamat"},
            "g2": {"nama": "Armored Core 4: For Answer", "tahun": "2008", "dev":
"Fromsoftware", "status": "Belum tamat"}
            }
        },

    "acc2": {
        "username": "Dapupu",
        "password": "Ketua_Nibung",
        "role": "member",
        "game": {
            "g1": {"nama": "MudRunner", "tahun": "2017", "dev": "Saber
Interactive", "status": "Belum Tamat"},
            }
        },

    "acc3": {
        "username": "Nabubu",
        "password": "Krotago",
        "role": "admin",
        "game": {}
        }
}
```

3.1.2. Menu Awal

Sebelum menggunakan program, pengguna harus login terlebih dahulu. Disini program juga memberikan menu register akun yang dapat digunakan pengguna untuk membuat akun terlebih dahulu jika tidak punya akun.

Source Code:

```
# Tampilan Header Program
print("=" * 80)
print("{:^80}".format("PROGRAM DIREKTORI GAME"))
print("=" * 80)

# Tampilan Menu
print("1 - Login Akun")
print("2 - Register Akun")
print("=" * 80)
```

```
# Tampilan Opsi Tambahan
print("0 - Akhiri Program")
print("=" * 80)

# Input Pilihan Pengguna
print("Silakan pilih menu yang anda inginkan:")
aksi = input("> ")
print("=" * 80)
```

3.1.3. Fitur Membersihkan Terminal

Fitur ini tersebar di seluruh kode program. Fitur ini berfungsi untuk membersihkan hasil output yang ada di terminal. Tujuannya adalah Terminal menjadi rapi dan memuat informasi yang benar-benar diperlukan oleh pengguna. Fitur ini hanya terdiri dari dua baris kode, yaitu *time.sleep()* dan *os.system()*. Kode *time.sleep()* berfungsi untuk menjeda program sesuai dengan waktu yang telah ditentukan sebelumnya, sedangkan kode *os.system()* berfungsi untuk menghapus semua output yang terdapat di terminal.

Tidak seperti di program sebelum, fitur ini ditambah satu baris kode dengan tujuan agar fitur ini bisa berjalan di *Operating System* yang berbeda.

Source Code:

```
def clear(): # Fungsi Untuk Membersihkan Terminal
    time.sleep(1)
    os.system('clear')
    os.system('cls')
```

3.1.4. Fitur Login

Source Code :

```
def login(): # Fungsi Untuk Login ke Program
    while True:
        try:
            global konfirmasi_login, kesempatan
            konfirmasi_login = False

            # Bagian Display Menu Login
            print("=" * 75)
            print(f"{'MENU LOGIN':^75}")
            print("=" * 75)
```

```
            username_login = input(f"{'Masukkan Username : ':<10}")
            password_login = input(f"{'Masukkan Password : ':<10}")

            if len(username_login) == 0 or len(password_login) == 0:
                raise ValueError ("Username atau Password tidak Boleh Kosong!")

            for id_akun, info in akun.items():
                if username_login == info['username'] and password_login ==
info['password']: # Bagian ini akan berjalan jika informasi yang dimasukkan
pengguna sama dengan informasi yang tersimpan di dictionary
                    global role_anda, id_anda, username_anda
                    konfirmasi_login = True # Mengubah status menjadi 'berhasil
login'
                    username_anda = info["username"]
                    role_anda = info['role'] # Mengambil role dari pengguna yang
login
                    id_anda = id_akun # Mengambil id akun dari pengguna yang
login

                    print("=" * 75)
                    print("Login Berhasil!")
                    print("=" * 75)
                    clear()

            if konfirmasi_login == True:
                break
            else:
                print("Login gagal, silahkan coba lagi!")
                kesempatan -= 1
                print(f"Sisa kesempatan : {kesempatan}")

                clear()

                if kesempatan == 0: # Memeriksa apakah kesempatan login pengguna
sudah habis
                    print("Kesempatan Login habis, silahkan anda membuat sebuah
akun!")

                    time.sleep(1)
                    clear()
                    register()
                    kesempatan = 4
                    break
        except ValueError as e:
            print("=" * 75)
            print(e)
            print("=" * 75)
            time.sleep(1)
            clear()

    return konfirmasi_login
```

3.1.5. Fitur Register

Source Code :

```
def register(username_baru, password_baru): # Fungsi untuk membuat sebuah akun
    while True:
        try:
            # Bagian yang memeriksa apakah ada elemen yang kosong
            if len(username_baru) and len(password_baru) != 0:
                print("=" * 60)
                print(f"{'KONFIRMASI AKUN BARU':^60}")
                print("=" * 60)

                print(f"{'Username':<20}: {username_baru}")
                print(f"{'Password':<20}: {password_baru}")

                # Bagian Konfirmasi akun
                print("=" * 60)
                print("Apakah informasi akun sudah benar? (y/n)")
                print("=" * 60)
                konfirmasi_regis = input("> ")

                clear()

                if konfirmasi_regis == "y": # Akan mendaftarkan aku ketika
pengguna mengetik 'y'
                    id_akun = f"acc{len(akun) + 1}"

                    akun[id_akun] = {
                        "username": username_baru,
                        "password": password_baru,
                        "role": "member",
                        "game": {}
                    }
                    print("=" * 60)
                    print("Akun berhasil dibuat!")
                    input("Tekan 'Enter' untuk Kembali")
                    print("=" * 60)
                    clear()
                    return akun

                elif konfirmasi_regis == "n": # Jika mengetik 'n' maka pengguna
akan mengulang proses registrasi
                    print("Mengulang Proses")
                    clear()
                else: # Akan berjalan jika pengguna salah input
                    print("Pilihan tidak valid")
                    clear()

            elif len(username_baru) == 0: # Akan berjalan jika ada elemen yang
```

```
kosong
                raise ValueError("Username Tidak Boleh Kosong")
            elif len(password_baru) == 0:
                raise ValueError("Password Tidak Boleh Kosong")
            elif len(username_baru) and len(password_baru) == 0:
                raise ValueError("Username dan Password Tidak boleh kosong")

        except ValueError as e:
            print(e)
            clear()
```

3.2. Fitur yang dimiliki oleh Admin dan User

3.2.1. Menu Direktori

Bagian ini merupakan penghubung dari setiap proses yang ada pada CRUD. Di bagian menu utama ini pengguna dapat memilih menu yang diinginkan.

Source Code:

```
# ... (Ada Kode dibagian sini)

clear()
konfirmasi_login = False # Mengatur apakah pengguna sudah login atau belum
(Sudah = True, Belum = False)
kesempatan = 4 # Mengatur kesempatan login yang dimiliki pengguna
gagal_login = False # Merupakan parameter yang akan digunakan apabila pengguna
menghabiskan kesempatan loginnya

# Tampilan Header Program
print("=" * 80)
print("{:^80}".format("PROGRAM DIREKTORI GAME"))
print("=" * 80)

# Tampilan Menu
print("1 - Login Akun")
print("2 - Register Akun")
print("=" * 80)

# Tampilan Opsi Tambahan
print("0 - Akhiri Program")
print("=" * 80)

# Input Pilihan Pengguna
print("Silakan pilih menu yang anda inginkan:")
aksi = input("> ")
```

```
print("=" * 80)
clear()

# ... (Ada Kode dibagian sini)
```

3.2.2. Fitur menambahkan game baru ke dalam direktori (Create)

Seperti yang sudah dijelaskan sebelumnya, pengguna dapat menambahkan game baru ke dalam direktori. Di bagian ini terdapat sebuah mekanisme yang memastikan agar data yang dimasukkan oleh pengguna tidak kosong. Berikut merupakan cuplikan singkat dari fitur ini beserta mekanisme pengaman agar data yang dimasukkan tidak kosong.

Source Code:

```
def tambah_game(): # Fungsi untuk menambahkan game baru ke direktori
    while True:
        try:
            # Bagian Display Menu
            print("=" * 74)
            print(f"{'MENAMBAHKAN GAME KE DIREKTORI':^74}")
            print("=" * 74)

            # Bagian Input game beserta beberapa elemen lainnya
            game_baru = input(f"{'Masukkan Judul Game':<35}: ")
            tahun_rilis = input(f"{'Masukkan Tahun Rilis Game':<35}: ")
            developer_game = input(f"{'Masukkan Developer Game':<35}: ")
            status_game = input(f"{'Masukkan Status Game':<35}: ")
            print("=" * 74)

# ... (Ada Kode dibagian sini)
```

3.2.3. Fitur melihat game yang terdapat pada direktori (Read)

Fitur ini cukup sederhana. Pengguna dapat melihat game apa saja yang terdapat di direktori.

Source code:

```
def lihat_game(): # Fungsi untuk melihat game yang terdapat di direktori
    # Bagian Display menu
    print("=" * 90)
    print(f"{'KOLEKSI GAME':^90}")
    print("=" * 90)
    print(f"{'ID':<5} {'Judul Game':<30} {'Tahun Rilis':<15} {'Developer':<20}
{'Status':<10}")
    for gid, game in akun[id_anda]['game'].items():
```

```
        print(f"{gid:<5} {game['nama']:<30} {game['tahun']:<15}
{game['dev']:<20} {game['status']:<10}")

    print("=" * 90)
    input("Tekan 'Enter' untuk kembali ke menu utama...")
    print("=" * 90)

    clear()

# ... (Ada Kode dibagian sini)
```

3.2.4. Fitur Mengganti data game yang diinginkan (Update)

Bagian ini memungkinkan pengguna untuk mengganti data game yang ada di direktori jika seandainya pengguna melakukan kesalahan pada saat proses *Create*. Sama seperti proses *Create*, bagian ini dilengkapi dengan mekanisme yang dapat memeriksa apakah pengguna memasukkan data kosong atau tidak. Berikut ini merupakan cuplikan kode yang berisi fitur tersebut.

Source code:

```
def update_game():
    while True:
        try:
            game_update_benar = False # Merupakan parameter yang melihat apakah
id game yang dimasukkan pengguna benar atau tidak
            # Display menu
            print("=" * 90)
            print(f"{'MENU UPDATE':^90}")
            print("=" * 90)
            print(f"{'ID':<5} {'Judul Game':<30} {'Tahun Rilis':<15}
{'Developer':<20} {'Status':<10}")
            for gid, game in akun[id_anda]['game'].items():
                print(f"{gid:<5} {game['nama']:<30} {game['tahun']:<15}
{game['dev']:<20} {game['status']:<10}")

            print("=" * 90)
            print(f"{'0':<5} Keluar")
            print("=" * 90)

            # Input ID game yang ingin diupdate
            print("Pilih ID Game yang Ingin Anda Update :")
            print("=" * 90)
            pilih_update = input("> ")

            clear()
```

```
# ... (Ada Kode dibagian sini)
```

3.2.5. Fitur Menghapus Game yang Sudah tidak diinginkan lagi (Delete)

Melalui fitur ini pengguna dapat menghapus game yang tidak diinginkan lagi dari direktori. Sifat dari proses *delete* ini adalah permanen, yang artinya data game akan terhapus selamanya jika pengguna memilih untuk menghapus data game tersebut. Oleh karena itu, program dilengkapi dengan sebuah pertanyaan yang bertujuan untuk memastikan kembali apakah pengguna yakin ingin menghapus game dari direktori.

Source Code:

```
def hapus_game(): # Fungsi untuk menghapus game yang dimiliki pengguna
    while True:
        try:
            selesai_delete = False # Merupakan parameter yang mengatur apakah
pengguna sudah selesai menghapus atau belum
            # Display Menu
            print("=" * 90)
            print(f"{'MENU HAPUS':^90}")
            print("=" * 90)
            print(f"{'ID':<5} {'Judul Game':<30} {'Tahun Rilis':<15}
{'Developer':<20} {'Status':<10}")
            for gid, game in akun[id_anda]['game'].items():
                print(f"{gid:<5} {game['nama']:<30} {game['tahun']:<15}
{game['dev']:<20} {game['status']:<10}")

            print("=" * 90)
            print(f"{'0':<5} Keluar")
            print("=" * 90)

            # Bagian input ID game yang ingin dihapus
            print("Masukkan ID Game yang ingin Anda hapus :")
            print("=" * 90)
            pilih_hapus = input("> ")

            clear()

# ... (Ada Kode dibagian sini)
```

3.3. Fitur yang hanya dimiliki oleh Admin

3.3.1. Menu Khusus Admin

Menu ini hanya dapat diakses jika akun memiliki role sebagai admin. Menu ini menampilkan beberapa fungsi yang dapat digunakan oleh admin untuk mengatur akun yang terdaftar di dictionary.

Source Code:

```
def menu_admin(): # Fungsi untuk memperlihatkan menu admin
    while True:
        try:
            # Display menu
            print("=" * 50)
            print(f"{'MENU ADMIN':^50}")
            print("=" * 50)

            print("1 - Tambahkan Akun")
            print("2 - Lihat Akun")
            print("3 - Update Info Akun")
            print("4 - Hapus Akun")
            print("=" * 50)

            print("0 - Keluar")
            print("=" * 50)

            print("Pilih Menu yang Anda Inginkan :")
            print("=" * 50)
            menu_admin = input("> ")

            clear()

# ... (Ada Kode dibagian sini)
```

3.3.2. Fitur menambahkan akun baru (Create)

Admin dapat menambahkan akun baru tanpa harus lewat menu register yang ada di menu utama. Source code fitur ini sama dengan source code *register* karena fitur ini hanya memanggil fungsi *register*.

3.3.3. Fitur melihat akun yang terdaftar di direktori (Read)

Fitur ini cukup sederhana sama dengan fitur Read yang ada pada koleksi game. Disini admin dapat melihat informasi akun yang terdaftar di direktori, mulai dari username, password serta role akun.

Source code:

```
def lihat_akun(): # Fungsi untuk melihat akun yang terdapat di direktori
    # Menampilkan semua informasi aku yang ada pada dictionary
    print("=" * 60)
    print(f"{'AKUN-AKUN YANG TERDAFTAR':^60}")
    print("=" * 60)

    print(f"{'ID':<5} {'Nama Akun':<20} {'Password':<20} {'Role':<10}")
    for id_acc, info_akun in akun.items():
        print(f"{str(id_acc):<5} {info_akun['username']:<20}
{info_akun['password']:<20} {info_akun['role']:<10}")

    print("=" * 60)
    input("Tekan 'Enter' untuk kembali ke Menu Admin... ")
    print("=" * 60)

    clear()
```

3.3.4. Fitur Mengganti informasi akun yang diinginkan (Update)

Bagian ini memungkinkan admin untuk mengganti informasi yang terdapat pada akun, mulai dari username, password serta role akun.

Source code:

```
def update_akun(): # Fungsi untuk mengupdate Informasi Akun
    while True:
        try:
            # Bagian ini berfungsi untuk mengganti Username, Password, dan juga
role dari akun-akun yang ada
            # Prinsip kerjanya kurang lebih sama dengan proses Update yang ada
pada menu "Update Informasi Game"
            print("=" * 70)
            print(f"{'UPDATE INFORMASI AKUN':^70}")
            print("=" * 70)

            print(f"{'ID':<5} {'Nama Akun':<25} {'Password':<25} {'Role'}")
            for id_acc, info_akun in akun.items():
                print(f"{str(id_acc):<5} {info_akun['username']:<25}
{info_akun['password']:<25} {info_akun['role']}")
```

```
            print("=" * 70)
            print(f"{'0':<5} Keluar")
            print("=" * 70)
            print("Masukkan ID Pengguna yang Ingin Anda Ubah:")
            admin_update = input("> ")
            print("=" * 70)

            clear()

# ... (Ada Kode dibagian sini)
```

3.3.5. Fitur Menghapus akun (Delete)

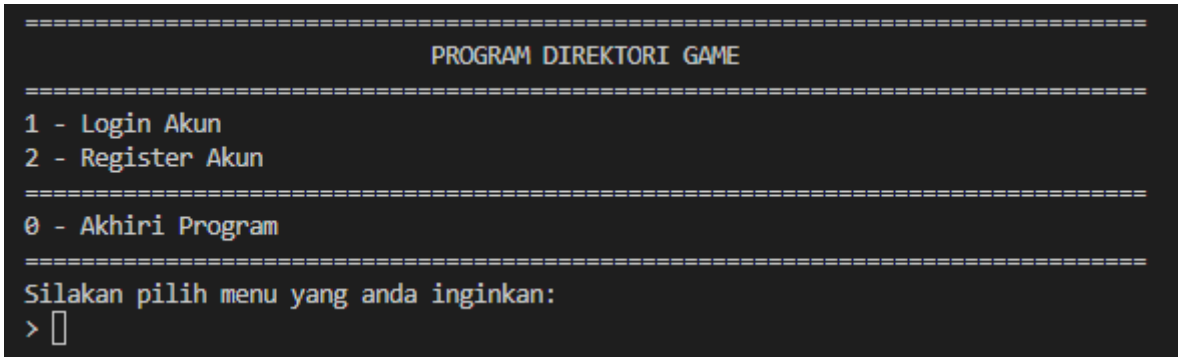
Melalui fitur ini pengguna dapat menghapus akun yang tidak diinginkan lagi. Sama seperti fitur Delete pada koleksi game, sifat dari proses *delete* ini adalah permanen, yang artinya data game akan terhapus selamanya jika pengguna memilih untuk menghapus data game tersebut. Oleh karena itu, program dilengkapi dengan sebuah pertanyaan yang bertujuan untuk memastikan kembali apakah pengguna yakin ingin menghapus game dari direktori.

Source Code:

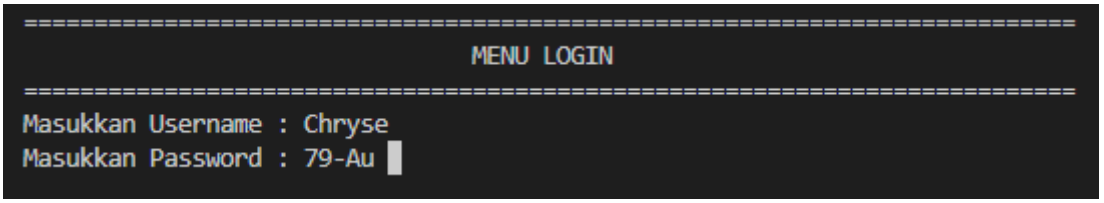
```
def hapus_akun(): # Fungsi untuk menghapus akun yang terdaftar
    while True:
        try:
            global akun
            selesai_delete = False
            print("=" * 70)
            print(f"{'HAPUS AKUN':^70}")
            print("=" * 70)

            print(f"{'ID':<5} {'Nama Akun':<25} {'Password':<25} {'Role'}")
            for id_acc, info_akun in akun.items():
                print(f"{str(id_acc):<5} {info_akun['username']:<25}
{info_akun['password']:<25} {info_akun['role']}")

            print("=" * 70)
            print(f"{'0':<5} Kembali")
            print("=" * 70)
            print("Masukkan ID akun yang ingin dihapus :")
            print("=" * 70)
            pilih_hapus = input("> ")
            print("=" * 70)

            clear()
```

```
# ... (Ada Kode dibagian sini)
```

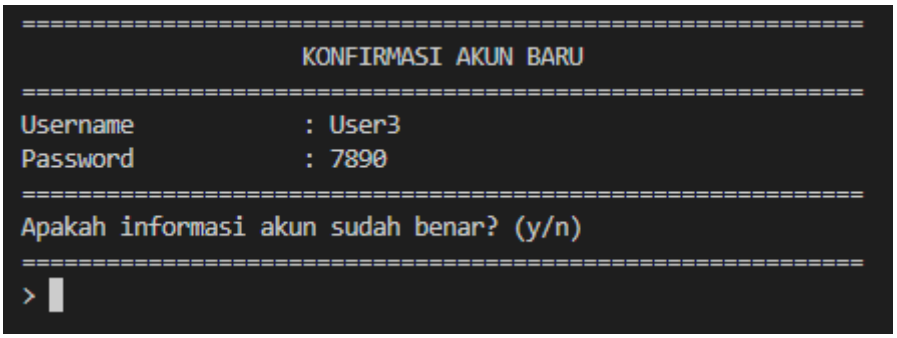
4. Hasil Output



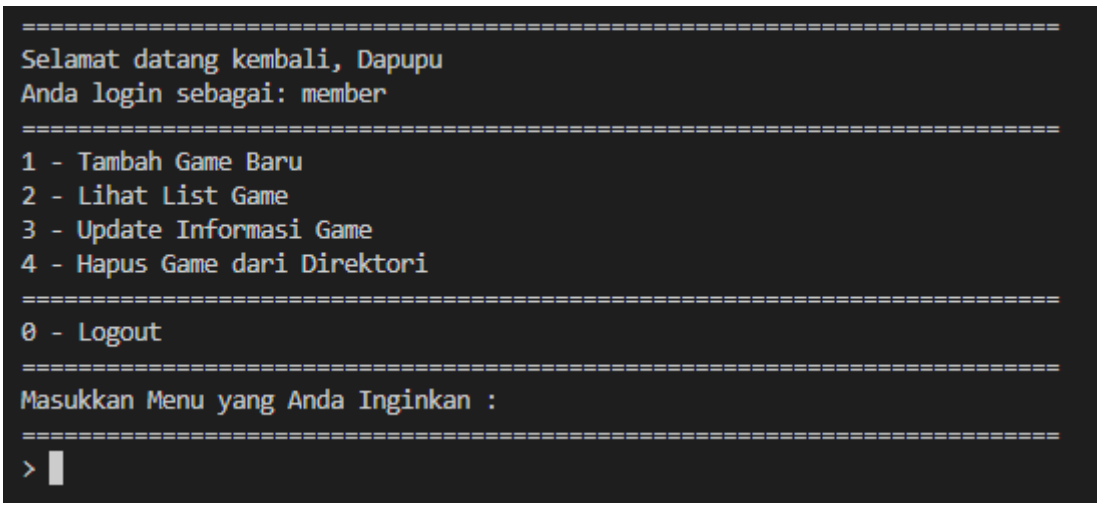
Gambar 4.1 Tampilan menu awal program



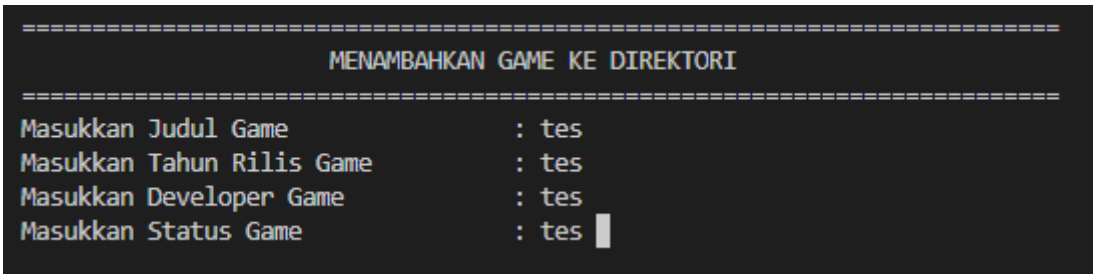
Gambar 4.2 Tampilan menu login



Gambar 4.3 Tampilan menu register



Gambar 4.4 Tampilan menu bagi admin



Gambar 4.5 Tampilan menu tambah game (Create) ke direktori

Gambar 4.6 Tampilan menu lihat (Read) daftar game

```
================================================================================
                                  MENU UPDATE
================================================================================
ID    Judul Game                  Tahun Rilis     Developer           Status
g1    MudRunner                   2017            Saber Interactive   Belum Tamat
g2    tes                         tes             tes                 tes
================================================================================
0     Keluar
================================================================================
Pilih ID Game yang Ingin Anda Update :
================================================================================
>
```

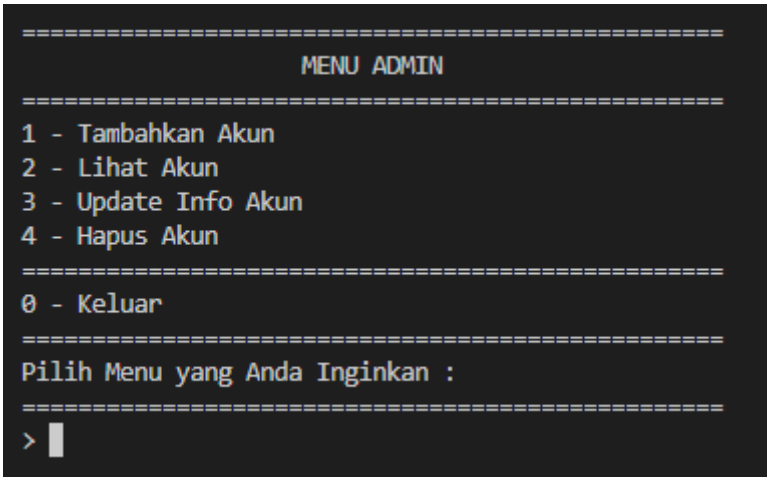
Gambar 4.7 Tampilan menu ubah (Update) game

```
================================================================================
                                  MENU HAPUS
================================================================================
ID    Judul Game                  Tahun Rilis     Developer           Status
g1    MudRunner                   2017            Saber Interactive   Belum Tamat
g2    tes                         tes             tes                 tes
================================================================================
0     Keluar
================================================================================
Masukkan ID Game yang ingin Anda hapus :
================================================================================
>
```

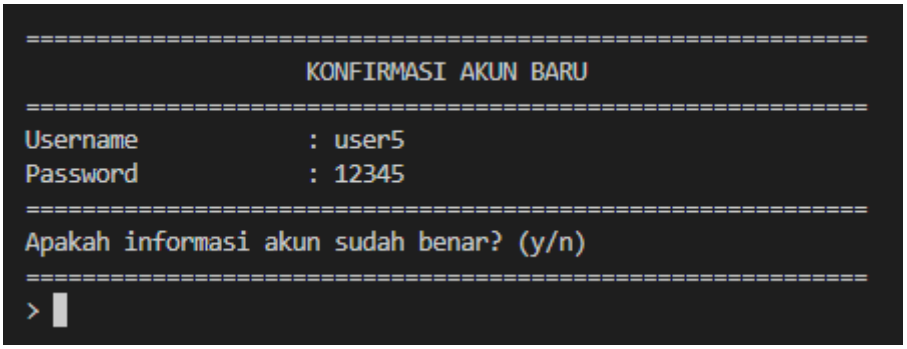
Gambar 4.8 Tampilan menu jika pengguna ingin menghapus game

```
================================================================================
Selamat datang kembali, Chryse
Anda login sebagai: admin
================================================================================
1 - Tambah Game Baru
2 - Lihat List Game
3 - Update Informasi Game
4 - Hapus Game dari Direktori
================================================================================
0 - Logout
5 - Menu Admin
================================================================================
Masukkan Menu yang Anda Inginkan :
================================================================================
>
```

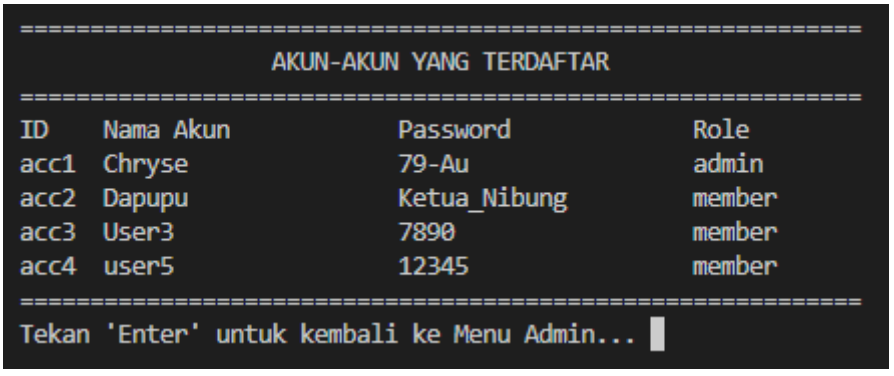
Gambar 4.9 Tampilan menu jika pengguna adalah admin



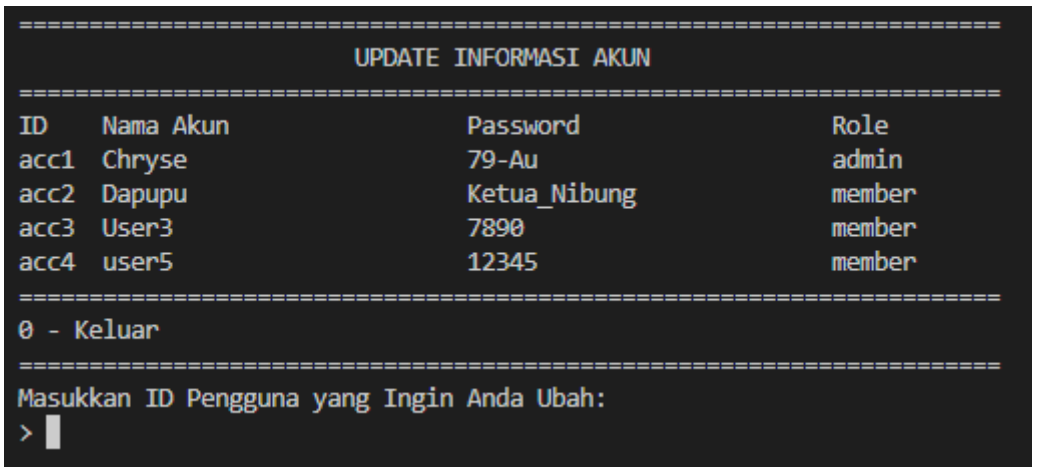
Gambar 4.10 Tampilan dari Menu Admin



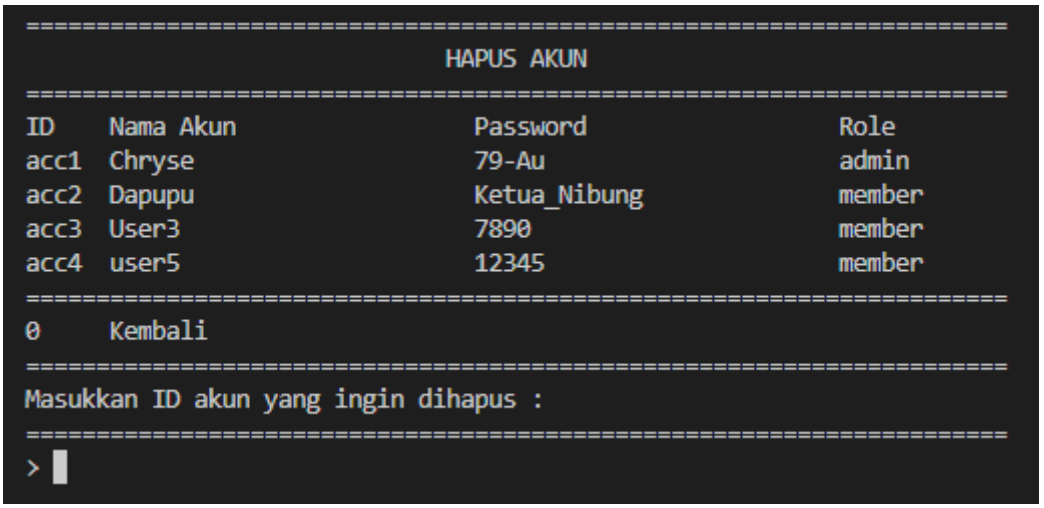
Gambar 4.11 Tampilan menu tambah akun (Read) bagi admin



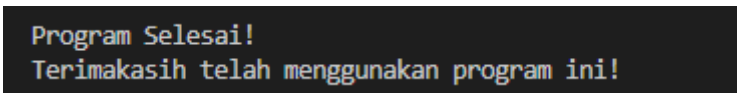
Gambar 4.12 Tampilan daftar akun (Read)



Gambar 4.13 Tampilan mengubah informasi (Update) akun



Gambar 4.14 Tampilan menghapus (Delete) akun



Gambar 4.15 Tampilan jika program telah diakhiri

5. Langkah-langkah GIT

5.1 GIT Add

```
PS E:\Git\praktikum-apd\post-test\post-test-apd-7> git add .
PS E:\Git\praktikum-apd\post-test\post-test-apd-7> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   2509106017-Yoga-Ananda-Prasetya-PT-7.py
        modified:   __pycache__/fungsi.cpython-313.pyc
        modified:   fungsi.py
```

Gambar 5.1 Git Add

"Git add" disini berfungsi untuk memasukkan file yang sudah kita kerjakan ke dalam staging area. Staging area disini berfungsi sebagai tempat singgah sementara file-file yang ingin kita commit di git nantinya. "Git status" disini hanya berfungsi untuk memeriksa apakah file yang ingin di-commit sudah ada di staging area atau belum.

5.2 GIT Commit

```
PS E:\Git\praktikum-apd\post-test\post-test-apd-7> git commit -m "Update Source Code untuk yang kedua kalinya + Dokumentasi"
[main 9741c51] Update Source Code untuk yang kedua kalinya + Dokumentasi
 3 files changed, 578 insertions(+), 503 deletions(-)
```

Gambar 5.2 Git Commit

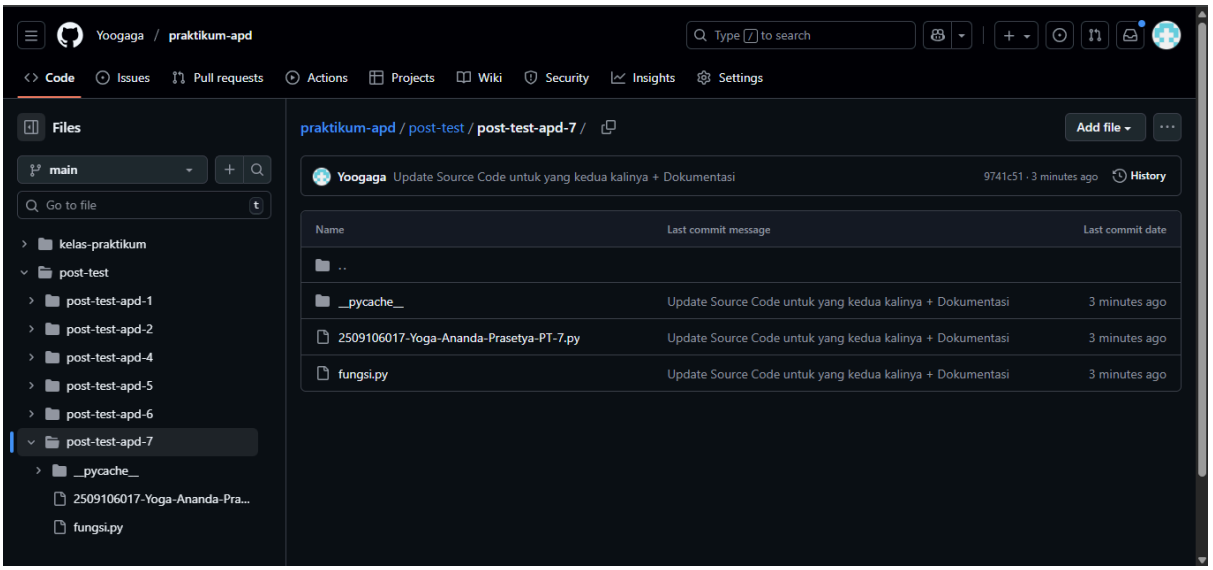
"Git Commit" disini berfungsi untuk menyimpan riwayat dari kode yang telah kita kerjakan ke dalam repository yang ada di dalam komputer kita (repository lokal). Disini kita juga dapat meninggalkan pesan singkat untuk mengingatkan kita atau orang lain terkait apa yang sudah kita kerjakan di kode tersebut.

5.3 GIT Push

```
PS E:\Git\praktikum-apd\post-test\post-test-apd-7> git push origin main
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 2 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 9.68 KiB | 762.00 KiB/s, done.
Total 8 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
remote: This repository moved. Please use the new location:
remote:   https://github.com/Yoogaga/praktikum-apd.git
To https://github.com/Yoogaga/belajargit.git
   ccbe355..9741c51  main -> main
```

Gambar 5.3 Git Push

"Git Push" disini berfungsi untuk mengunggah file-file yang sudah di-commit di repository lokal menuju ke repository yang ada di Github. Tujuannya adalah agar file-file yang sudah dikerjakan dapat diakses secara online dan dapat diakses oleh siapapun selama pengaturan repository-nya tidak di-private.



Gambar 5.4 Tampilan pada Github