LAPORAN PRAKTIKUM

POSTTEST 4

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Elfin Sinaga (2409106024)

Kelas (A2 ‘24)

PROGRAM STUDI INFORMATIKA

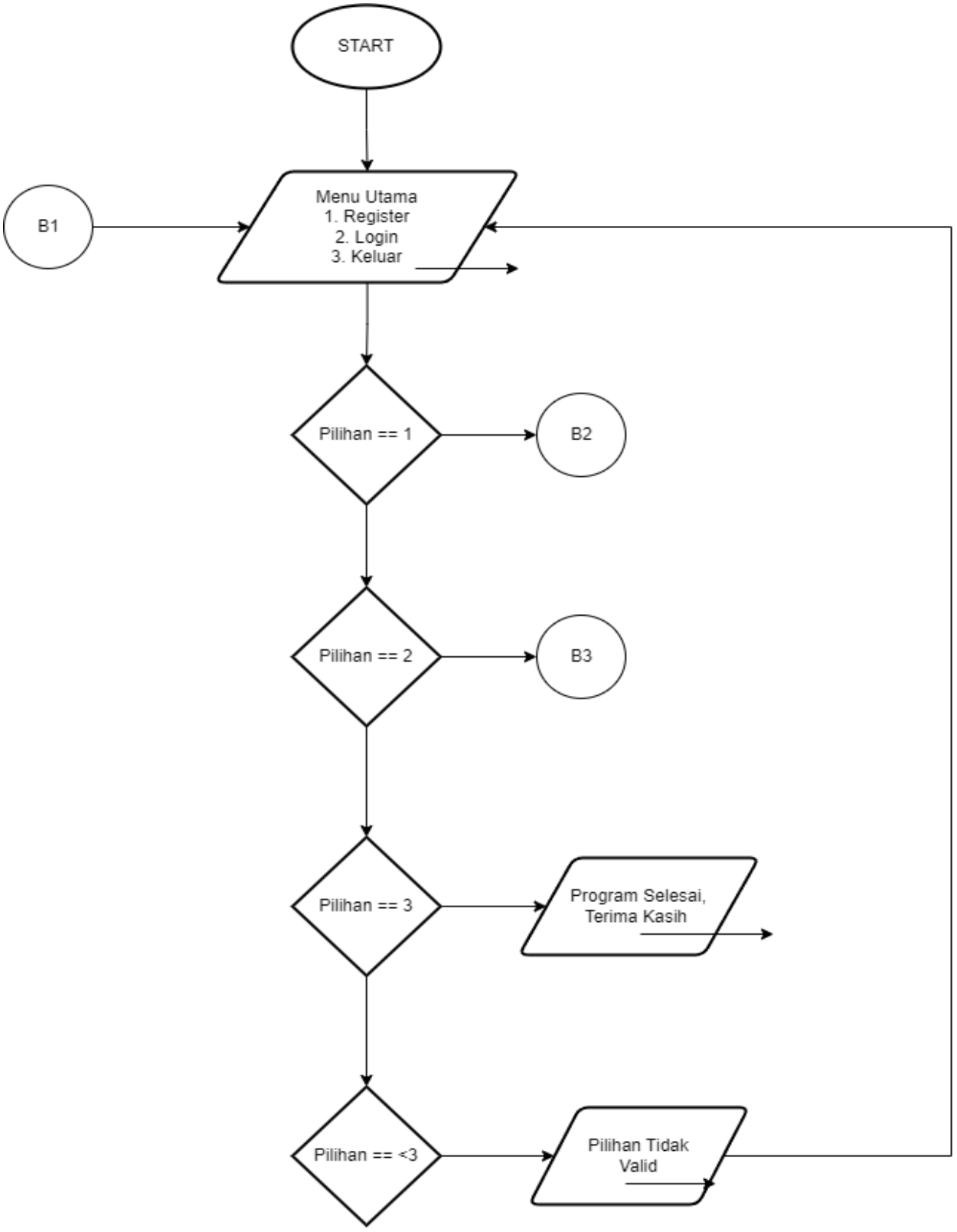
UNIVERSITAS MULAWARMAN

SAMARINDA

2025

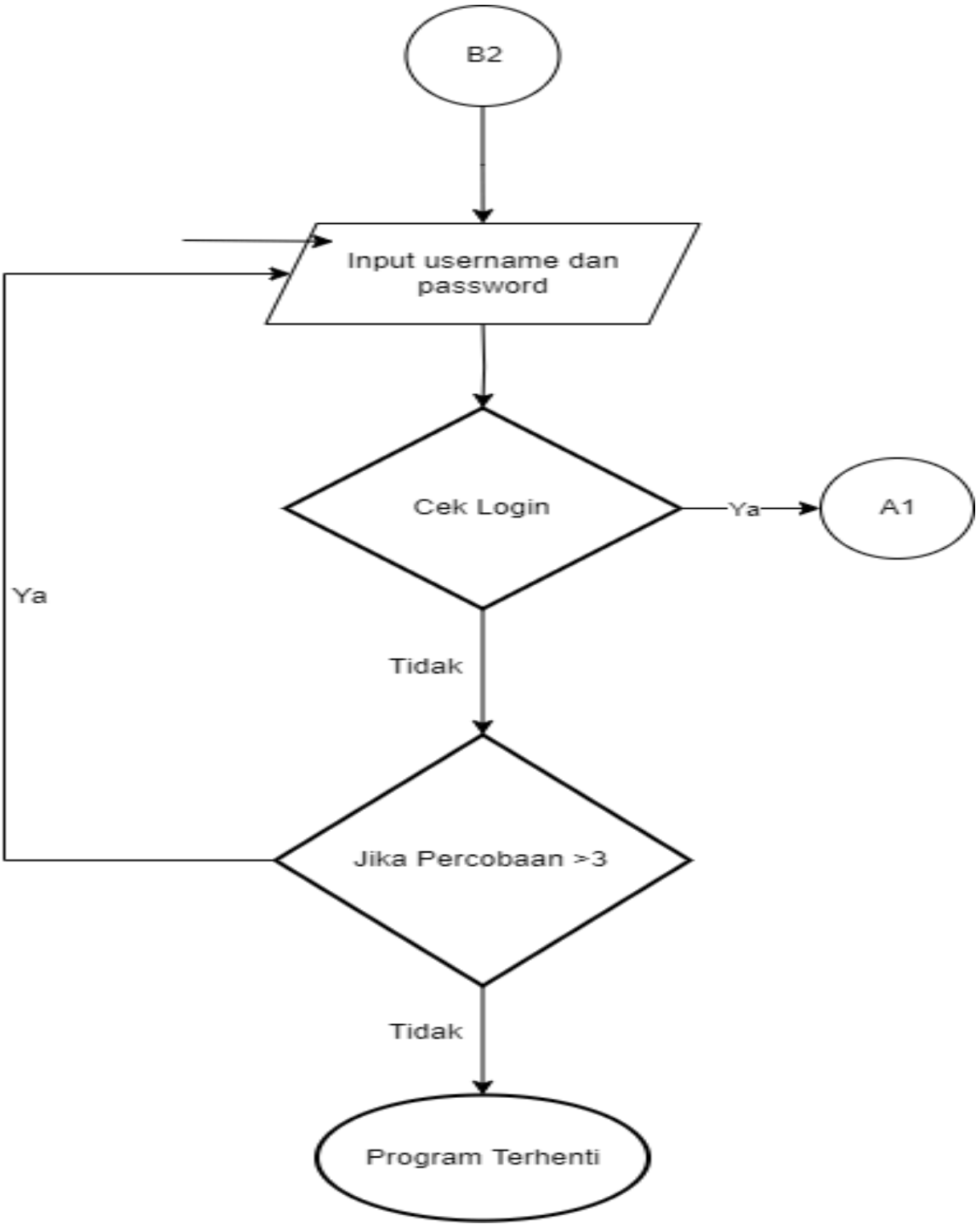
1. Flowchart

1.1 Menu Utama



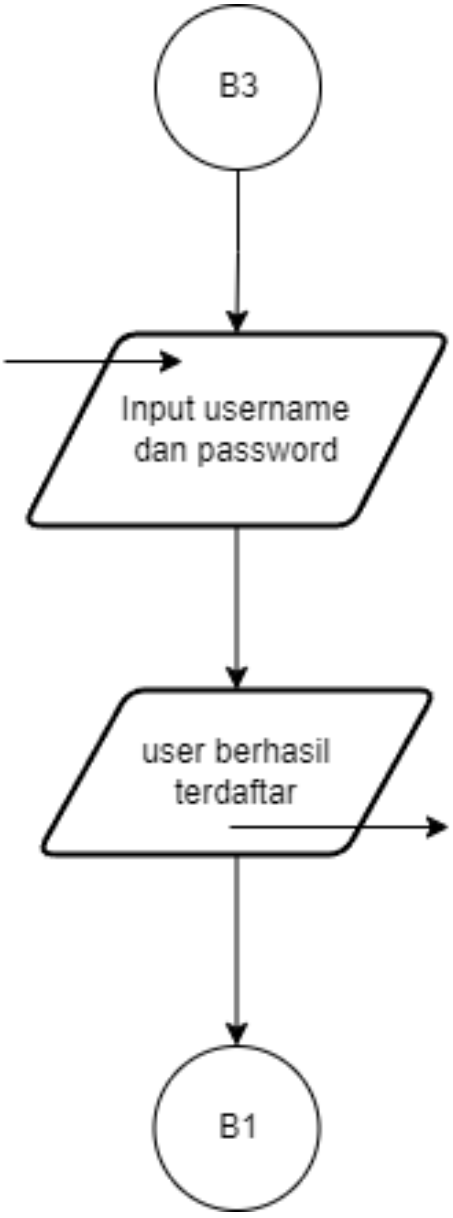
Gambar 1.1

1.2 Menu Login



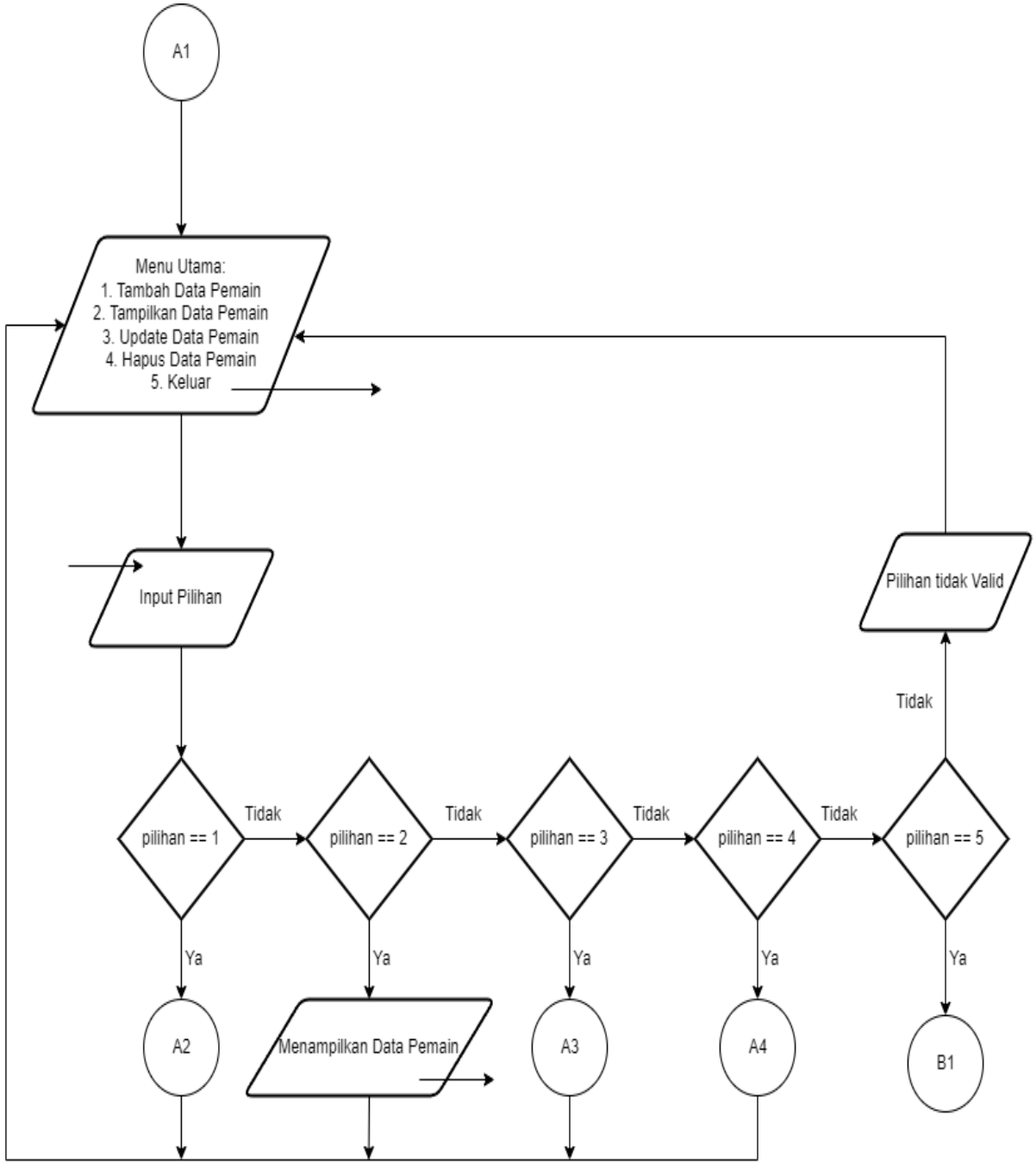
Gambar 1.2

1.3 Menu Register



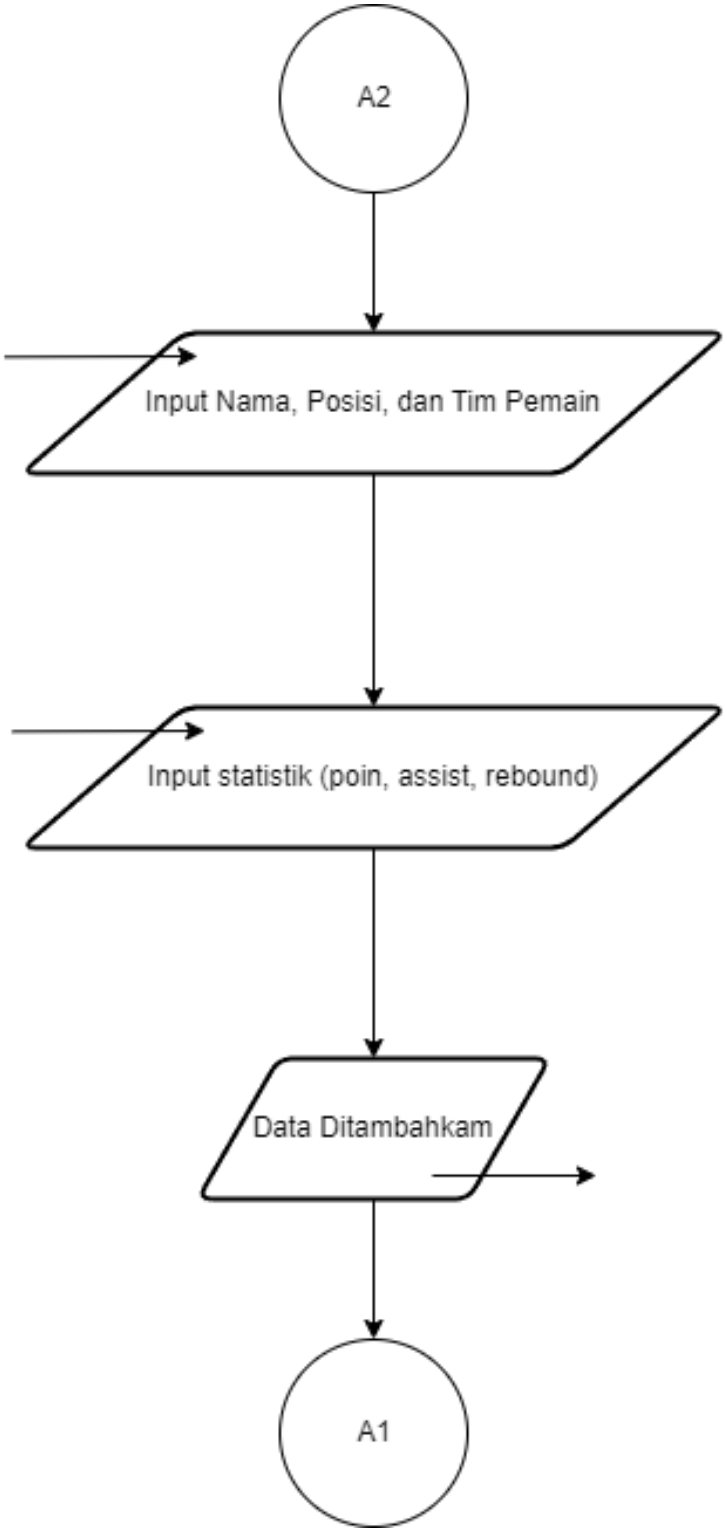
Gambar 1.3

1.4 Menu Tampilan



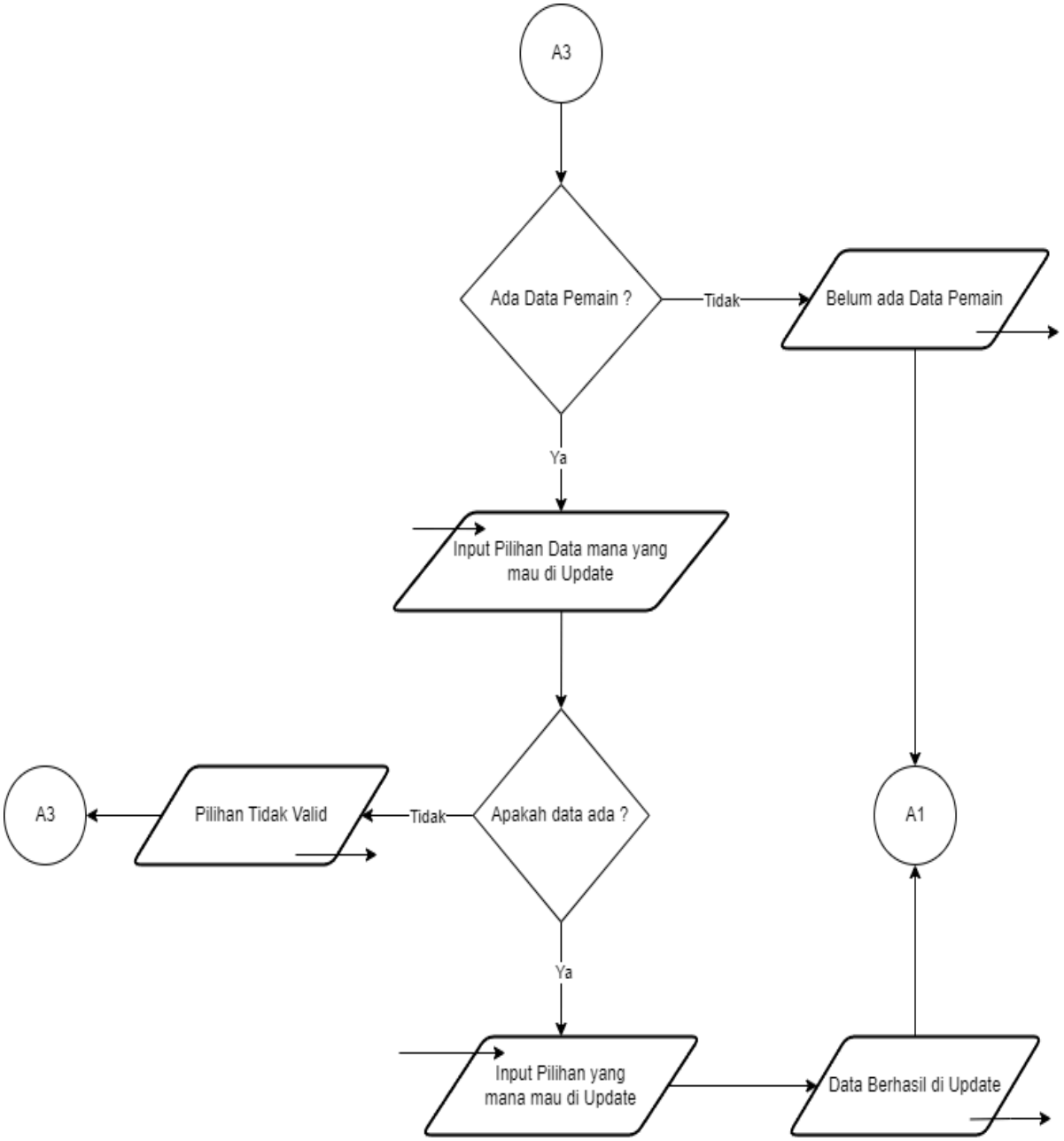
Gambar 1.4

1.5 Tambah Data



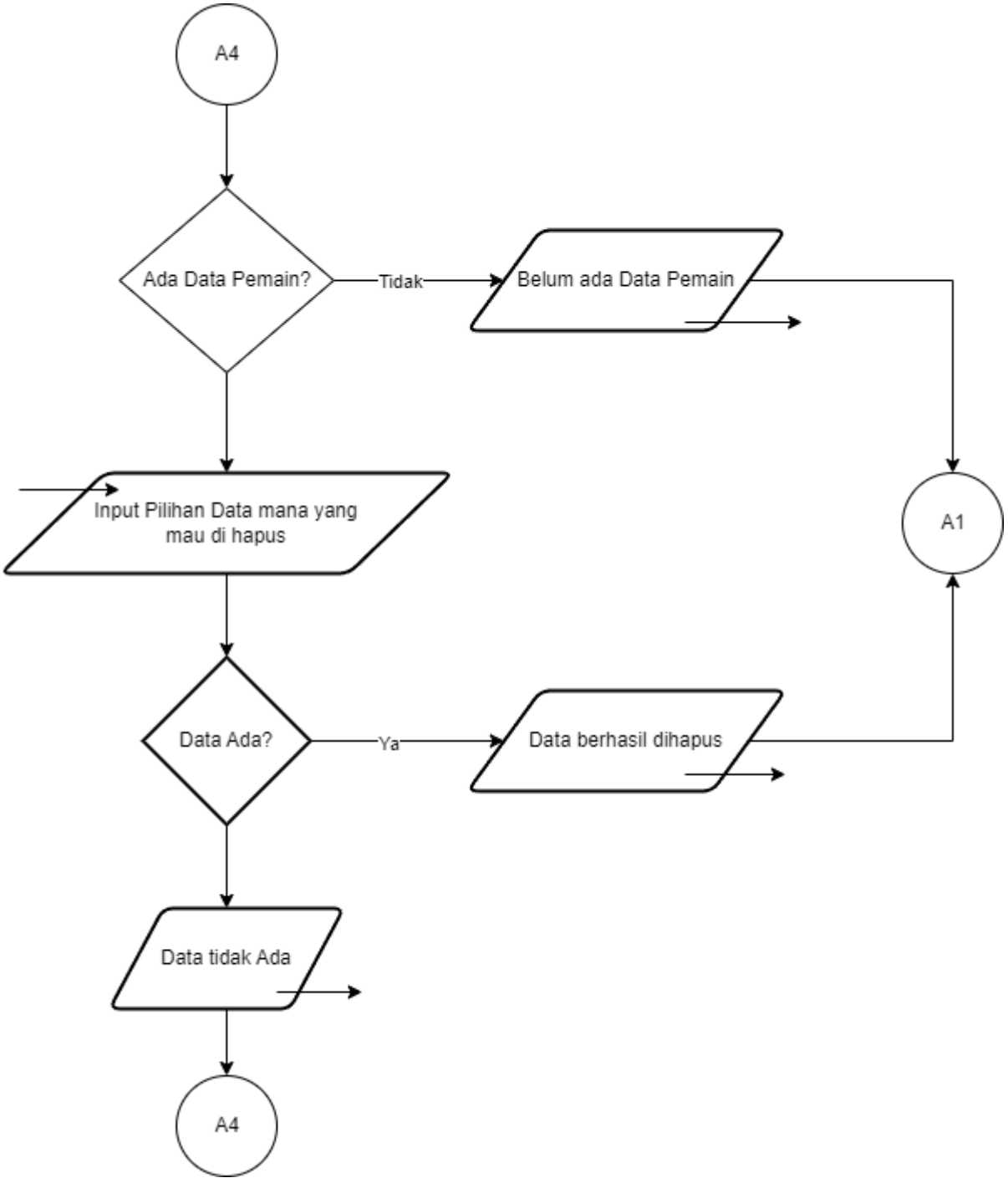
Gambar 1.5

1.6 Update Data



Gambar 1.6

1.7 Hapus Data



Gambar 1.7

2. Deskripsi Singkat Program

Program ini adalah aplikasi berbasis C++ untuk mengelola data pemain NBA. Pengguna dapat mendaftar dan login untuk mengakses fitur-fitur seperti:
1. Menambah data pemain (nama, posisi, tim, dan statistik poin, assist, rebound).
2. Menampilkan data pemain yang sudah tersimpan.
3. Mengupdate data pemain (nama, posisi, tim, dan statistik).
4. Menghapus data pemain dari data.

Program menggunakan array untuk menyimpan data pengguna dan pemain dengan kapasitas terbatas. Fitur login memiliki batasan 3 percobaan untuk keamanan.

3. Source Code

A. Menu Utama

```
    do {
        tampilkanHeader("Menu Utama");
        cout << "1. Register\n";
        cout << "2. Login\n";
        cout << "3. Keluar\n";
        cout << "Pilih menu: ";
        cin >> pilihan;

        switch (pilihan) {
            case 1:
                registerUser(users, jumlahUser, MAKS_USER);
                break;
            case 2:
                indeksloginuser = loginUser(users, jumlahUser,
PERCOBAAN_LOGIN_MAKS);
                if (indeksloginuser != -1) {
                    menuManajemenPemain(indeksloginuser);
                }
                break;
            case 3:
                cout << "Program selesai. Terima kasih!\n";
                break;
            default:
                cout << "Pilihan tidak valid. Silakan coba lagi.\n";
                break;
        }
    } while (pilihan != 3);
}
```

B. Register

```
bool registerUser(User users[], int &jumlahUser, int maxUser) {
    if (jumlahUser < maxUser) {
        tampilkanHeader("Register User Baru");
        cout << "Masukkan Username: ";
        cin >> users[jumlahUser].username;
        cout << "Masukkan Password: ";
        cin >> users[jumlahUser].password;
        jumlahUser++;
        cout << "User berhasil terdaftar!\n";
        return true;
    } else {
        cout << "Kapasitas user penuh. Tidak bisa mendaftarkan user baru.\n";
        return false;
    }
}
```

C. Login

```
int loginUser(User users[], int jumlahUser, int maxAttempts) {
    string username, password;
    int percobaan = 0;

    while (percobaan < maxAttempts) {
        tampilkanHeader("Login");
        cout << "Masukkan Username: ";
        cin >> username;
        cout << "Masukkan Password: ";
        cin >> password;

        for (int i = 0; i < jumlahUser; i++) {
            if (users[i].username == username && users[i].password == password) {
                cout << "Login berhasil! Selamat datang, " << username << "!\n";
                return i;
            }
        }

        percobaan++;
        cout << "Login gagal. Percobaan " << percobaan << " dari " << maxAttempts
<< ".\n";
    }

    cout << "Anda telah melebihi batas percobaan login. Program berhenti.\n";
    return -1;
```

```
}
```

D. Menu Manajemen

```
    do {
        tampilkanHeader("Manajemen Pemain NBA");
        cout << "1. Tambah Data Pemain\n";
        cout << "2. Tampilkan Data Pemain\n";
        cout << "3. Update Data Pemain\n";
        cout << "4. Hapus Data Pemain\n";
        cout << "5. Logout\n";
        cout << "Pilih menu: ";
        cin >> pilihanSetelahLogin;

        switch (pilihanSetelahLogin) {
            case 1:
                tambahDataPemain(dataPemain, jumlahPemain, MAKS_PEMAIN);
                break;
            case 2:
                tampilkanDataPemain(dataPemain, jumlahPemain);
                break;
            case 3:
                updateDataPemain(dataPemain, jumlahPemain);
                break;
            case 4:
                hapusDataPemain(dataPemain, jumlahPemain);
                break;
            case 5:
                cout << "Anda telah logout.\n";
                indeksloginuser = -1;
                break;
            default:
                cout << "Pilihan tidak valid.\n";
                break;
        }
    } while (indeksloginuser != -1);
}
```

E. Tambah Data

```
bool tambahDataPemain(Pemain dataPemain[], int &jumlahPemain, int maxPemain) {
    if (jumlahPemain < maxPemain) {
        tampilkanHeader("Tambah Data Pemain");
        cin.ignore();
        cout << "Masukkan Nama Pemain: ";
        getline(cin, dataPemain[jumlahPemain].nama);
```

```
        cout << "Masukkan Posisi Pemain: ";
        getline(cin, dataPemain[jumlahPemain].posisi);
        cout << "Masukkan Tim Pemain: ";
        getline(cin, dataPemain[jumlahPemain].tim);
        cout << "Masukkan Statistik Pemain (poin assist rebound)\n";
        cout << "Poin : ";
        cin >> dataPemain[jumlahPemain].statistik.poin;
        cout << "Assist : ";
        cin >> dataPemain[jumlahPemain].statistik.assist;
        cout << "Rebound : ";
        cin >> dataPemain[jumlahPemain].statistik.rebound;
        jumlahPemain++;
        cout << "Data pemain berhasil ditambahkan.\n";
        return true;
    } else {
        cout << "Kapasitas pemain penuh. Tidak bisa menambah data lagi.\n";
        return false;
    }
}
```

F. Tampilkan Data

```
void tampilkanDataPemain(Pemain dataPemain[], int jumlahPemain) {
    if (jumlahPemain == 0) {
        cout << "\nBelum ada data pemain.\n";
        return;
    }

    tampilkanHeader("Data Pemain NBA");
    cout << "-------------------------------------\n";
    for (int i = 0; i < jumlahPemain; i++) {
        cout << "Pemain ke-" << i + 1 << ":\n";
        cout << "Nama: " << dataPemain[i].nama << "\n";
        cout << "Posisi: " << dataPemain[i].posisi << "\n";
        cout << "Tim: " << dataPemain[i].tim << "\n";
        cout << "Statistik:\n";
        cout << "  Poin: " << dataPemain[i].statistik.poin << "\n";
        cout << "  Assist: " << dataPemain[i].statistik.assist << "\n";
        cout << "  Rebound: " << dataPemain[i].statistik.rebound << "\n";
        cout << "  Total Statistik: " <<
hitungTotalStatistik(dataPemain[i].statistik) << "\n";
        cout << "-------------------------------------\n";
    }
}
```

G. Update Data

```
bool updateDataPemain(Pemain dataPemain[], int jumlahPemain) {
    if (jumlahPemain == 0) {
        cout << "\nBelum ada data pemain untuk diupdate.\n";
        return false;
    }

    tampilkanHeader("Update Data Pemain");
    int index;
    cout << "Masukkan nomor pemain yang ingin diupdate (1-" << jumlahPemain << "):
";
    cin >> index;

    if (index > 0 && index <= jumlahPemain) {
        int pilihanUpdate;
        cout << "\nPilih data yang ingin diupdate:\n";
        cout << "1. Nama\n";
        cout << "2. Posisi\n";
        cout << "3. Tim\n";
        cout << "4. Statistik\n";
        cout << "Pilihan: ";
        cin >> pilihanUpdate;

        cin.ignore();
        switch (pilihanUpdate) {
            case 1:
                cout << "Masukkan Nama Pemain Baru: ";
                getline(cin, dataPemain[index - 1].nama);
                cout << "Nama pemain berhasil diupdate.\n";
                break;
            case 2:
                cout << "Masukkan Posisi Pemain Baru: ";
                getline(cin, dataPemain[index - 1].posisi);
                cout << "Posisi pemain berhasil diupdate.\n";
                break;
            case 3:
                cout << "Masukkan Tim Pemain Baru: ";
                getline(cin, dataPemain[index - 1].tim);
                cout << "Tim pemain berhasil diupdate.\n";
                break;
            case 4:
                cout << "Masukkan Statistik Pemain Baru (poin assist rebound)\n";
                cout << "Poin : ";
                cin >> dataPemain[index - 1].statistik.poin;
```

```
                cout << "Assist : ";
                cin >> dataPemain[index - 1].statistik.assist;
                cout << "Rebound : ";
                cin >> dataPemain[index - 1].statistik.rebound;
                cout << "Statistik pemain berhasil diupdate.\n";
                break;
            default:
                cout << "Pilihan tidak valid.\n";
                return false;
        }
        return true;
    } else {
        cout << "Nomor pemain tidak valid.\n";
        return false;
    }
}
```

H. Hapus Data

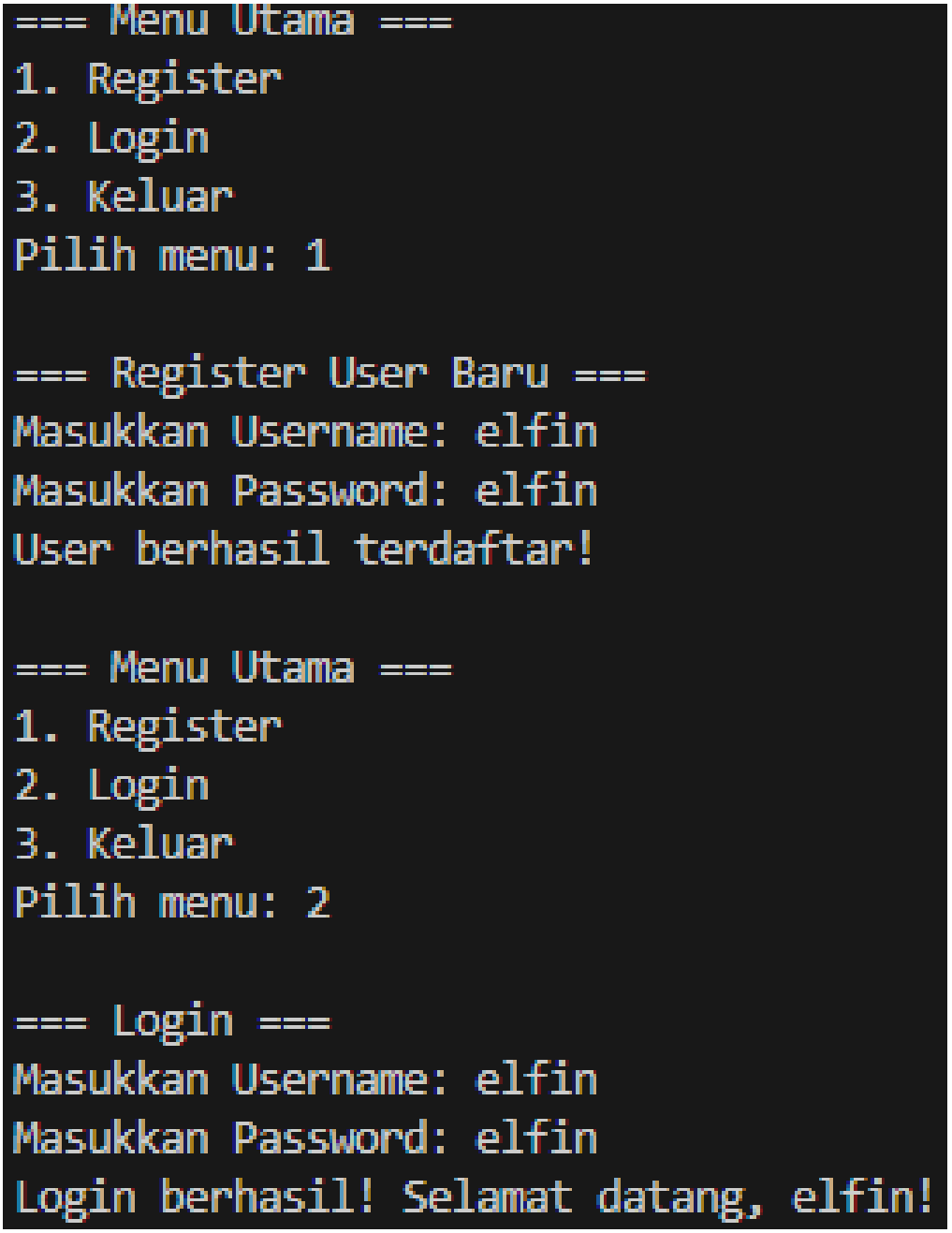
```
bool hapusDataPemain(Pemain dataPemain[], int &jumlahPemain) {
    if (jumlahPemain == 0) {
        cout << "\nBelum ada data pemain untuk dihapus.\n";
        return false;
    }

    tampilkanHeader("Hapus Data Pemain");
    int index;
    cout << "Masukkan nomor pemain yang ingin dihapus (1-" << jumlahPemain << "):
";
    cin >> index;

    if (index > 0 && index <= jumlahPemain) {
        for (int i = index - 1; i < jumlahPemain - 1; i++) {
            dataPemain[i] = dataPemain[i + 1];
        }
        jumlahPemain--;
        cout << "Data pemain berhasil dihapus.\n";
        return true;
    } else {
        cout << "Nomor pemain tidak valid.\n";
        return false;
    }
}
```

I. Mendeklarasikan Fungsi

```
void tampilkanHeader(string judul);
bool registerUser(User users[], int &jumlahUser, int maxUser);
int loginUser(User users[], int jumlahUser, int maxAttempts);
void tampilkanDataPemain(Pemain dataPemain[], int jumlahPemain);
bool tambahDataPemain(Pemain dataPemain[], int &jumlahPemain, int maxPemain);
bool updateDataPemain(Pemain dataPemain[], int jumlahPemain);
bool hapusDataPemain(Pemain dataPemain[], int &jumlahPemain);
void menuUtama();
void menuManajemenPemain(int &indeksloginuser);
int hitungTotalStatistik(const Stat &stat);
```

4. Hasil Output

A. Menu Utama, Register, dan Login



Gambar 2.1

B. Menu Manajemen dan Menu Tambah Data

```
=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 1

=== Tambah Data Pemain ===
Masukkan Nama Pemain: Stephen Curru
Masukkan Posisi Pemain: Point Guard
Masukkan Tim Pemain: Golden State Warriors
Masukkan Statistik Pemain (poin assist rebound)
Poin : 235
Assist : 64
Rebound : 12
Data pemain berhasil ditambahkan.
```

Gambar 2.2

C. Tampilkan Data

```
=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 2

=== Data Pemain NBA ===
--------------------------------------
Pemain ke-1:
Nama: Stephen Curru
Posisi: Point Guard
Tim: Golden State Warriors
Statistik:
  Poin: 235
  Assist: 64
  Rebound: 12
  Total Statistik: 311
--------------------------------------
```

Gambar 2.3

D. Update Data

```
=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 3

=== Update Data Pemain ===
Masukkan nomor pemain yang ingin diupdate (1-1): 1

Pilih data yang ingin diupdate:
1. Nama
2. Posisi
3. Tim
4. Statistik
Pilihan: 3
Masukkan Tim Pemain Baru: Mavericks Dallas
Tim pemain berhasil diupdate.

=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 2

=== Data Pemain NBA ===
--------------------------------------
Pemain ke-1:
Nama: Stephen Curru
Posisi: Point Guard
Tim: Mavericks Dallas
Statistik:
  Poin: 235
  Assist: 64
  Rebound: 12
  Total Statistik: 311
--------------------------------------
```

Gambar 2.4

E. Hapus Data

```
=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 4

=== Hapus Data Pemain ===
Masukkan nomor pemain yang ingin dihapus (1-1): 1
Data pemain berhasil dihapus.

=== Manajemen Pemain NBA ===
1. Tambah Data Pemain
2. Tampilkan Data Pemain
3. Update Data Pemain
4. Hapus Data Pemain
5. Logout
Pilih menu: 2

Belum ada data pemain.
```

Gambar 2.5

5. Langkah Langkah Git

```
PS C:\Kuliah\praktikum-apl> git add .
PS C:\Kuliah\praktikum-apl> git commit -m "semoga nilai bagus"
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Kuliah\praktikum-apl> git push origin main
Everything up-to-date
PS C:\Kuliah\praktikum-apl>
```

1. Buka Terminal pada Visual Studio Code.
2. input “git add .” untuk menambahkan semua perubahan yang ada di direktori.
3. input “git commit -m” untuk menyimpan perubahan yang telah dilakukan pada repository lokal.
4. input “git push origin main” untuk mengunggah (push) commit yang telah dilakukan di branch lokal main ke branch main di repository remote.