BAHAN PRESENTASI KELOMPOK
LAPORAN HASIL PEMBUATAN PROGRAM DATA

Disusun Oleh :

I Komang Dika Gus Septa (42530015)

Friendly Rianta Dwi Pratama (42530033)

Dosen :

Ni Luh Putu Ika Candrawengi S.Stat., M.Stat

PROGRAM STUDI TEKNOLOGI INFORMASI

FAKULTAS TEKNIK DAN INFORMATIKA

UNIVERSITAS PENDIDIKAN NASIONAL

2025

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Coding adalah proses menulis serangkaian perintah atau instruksi ke dalam bahasa pemrograman agar komputer dapat melakukan tugas tertentu. Instruksi tersebut ditulis dalam bentuk kode, yang memiliki aturan atau sintaks khusus yang membuat komputer dapat memahami dan menjalankannya. *Coding* memungkinkan seseorang untuk mengontrol cara sebuah program beroperasi, mulai dari menerima *input*, memproses data, dan menghasilkan *output*. Berbagai bidang teknologi menggunakan *coding*, termasuk pembuatan aplikasi, *website*, *game*, pengolahan data, sistem informasi, dan kecerdasan buatan. Seorang programmer tidak hanya menulis kode, tetapi juga berpikir logis dan sistematis untuk menyelesaikan masalah selama proses *coding*. *Coding* adalah keterampilan penting di era teknologi saat ini karena membantu mengubah ide atau kebutuhan menjadi solusi digital yang dapat digunakan oleh komputer.

Python adalah bahasa pemrograman tingkat tinggi yang bersifat *open source* yang mudah dipelajari dan dapat digunakan di berbagai sistem operasi. Sintaksnya yang sederhana dan indentasi membuat programnya mudah dibaca dan dipahami oleh pemula maupun profesional. *Python* mendukung berbagai tipe data dan paradigma pemrograman, termasuk berorientasi objek dan prosedural. *Python* banyak digunakan di berbagai industri dan bidang pendidikan. *Python* digunakan dalam statistika dan analisis data untuk mengolah dan menganalisis data dengan berbagai *library* pendukung. Pengembangan *web*, kecerdasan buatan, pembelajaran mesin, dan otomasi sistem juga menggunakan *Python*. *Python* memiliki banyak kelebihan,

termasuk kemudahan penggunaan, fleksibilitas, dan banyak *library*. Kekurangannya, bagaimanapun, adalah kecepatan eksekusi yang agak lambat dibandingkan dengan beberapa bahasa pemrograman lain.

1.2 Rumusan Masalah

1. Apa Pengertian Manajemen Data?
2. Apa Konsep Dasar Basis Data?
3. Apa Kebutuhan Fungsionalnya?
4. Apa Kebutuhan Non-fungsionalnya?
5. Bagaimana Penjelasan Struktur Program?
6. Bagaimana Penjelasan Logika Khususnya?
7. Bagaimana *Flowchart* (Diagram Alur) dari Manajemen Data Mahasiswa?
8. Bagaimana Hasil *output*nya?

1.3 Tujuan Penulisan

1. Menjelaskan Pengertian Manajemen Data?
2. Menjelaskan Konsep Dasar Basis Data.
3. Menjelaskan Apa Saja Kebutuhan Fungsionalnya.
4. Menjelaskan Apa Saja Kebutuhan Non-fungsinalnya.
5. Menjelaskan Struktur Programnya.
6. Menjelaskan Logika Khususnya.
7. Menjelaskan *Flowchart* (Diagram Alur) dari Manajemen Data Mahasiswa.
8. Menjelaskan Hasil *Output*.

BAB II
PEMBAHASAN

2.1 Pengertian Manajemen Data

Manajemen data adalah proses sistematis untuk mengelola data mulai dari tahap pengumpulan, penyimpanan, pengolahan, pengorganisasian, hingga pemeliharaan dan pengamanan data agar dapat digunakan secara efektif dan efisien. Tujuan utama manajemen data adalah memastikan data yang dimiliki akurat, konsisten, mudah diakses, dan relevan untuk mendukung pengambilan keputusan.

Dengan manajemen data yang baik, sebuah organisasi atau individu dapat mengolah data menjadi informasi yang berguna, meningkatkan kinerja, dan membantu perencanaan dan evaluasi dengan lebih baik. Manajemen data juga mencakup pengaturan struktur data, pengendalian kualitas data, pengelolaan basis data, dan perlindungan data dari kehilangan atau penyalahgunaan.

2.2 Konsep Dasar Basis Data

Basis data, juga dikenal sebagai *database*, berfungsi sebagai tempat penyimpanan data yang saling berhubungan dan disimpan secara sistematis untuk memenuhi kebutuhan informasi. Konsep dasar basis data terdiri dari sekumpulan prinsip dan komponen utama yang digunakan untuk menyimpan, mengelola, dan mengakses data secara terstruktur sehingga mudah digunakan dan dikelola.

Basis data dasar terdiri dari data sebagai fakta mentah, tabel sebagai tempat penyimpanan data dalam bentuk baris (catatan) dan kolom (*field*), dan relasi yang menunjukkan hubungan antar tabel. Selain itu, ada kunci utama yang

berfungsi sebagai penanda unik untuk setiap data, dan kunci luar yang berfungsi sebagai kunci yang menghubungkan antar tabel. Untuk mengelola basis data, *DBMS* (*Database Management System*) adalah perangkat lunak yang memungkinkan pengguna membuat, mengubah, menghapus, dan mengamankan data. Memahami konsep dasar basis data memungkinkan pengelolaan informasi dilakukan secara lebih efektif, akurat, dan terintegrasi.

2.3 Kebutuhan Fungsional

Kebutuhan fungsional manajemen data mahasiswa adalah fungsi-fungsi utama yang harus dimiliki oleh sistem agar dapat mengelola data mahasiswa secara efektif dan sesuai kebutuhan akademik. Berikut kebutuhan fungsionalnya:

2.3.1 Pengelolaan Data Mahasiswa

Sistem mampu menambah, mengubah, menghapus, dan menampilkan data mahasiswa seperti NIM, nama, program studi, angkatan, dan status mahasiswa.

2.3.2 Pencarian dan Penyaringan Data

Sistem menyediakan fitur pencarian data mahasiswa berdasarkan kriteria tertentu, seperti NIM, nama, program studi, atau angkatan.

2.3.3 Pengelolaan Data Akademik

Sistem dapat mengelola data akademik mahasiswa, seperti mata kuliah yang diambil, nilai, IPK, dan riwayat studi.

2.3.4 Validasi dan Konsistensi Data

Sistem melakukan validasi untuk memastikan data yang dimasukkan lengkap, benar, dan tidak duplikat, khususnya pada data penting seperti NIM.

2.3.5 Penyimpanan dan Keamanan Data

Sistem menyimpan data mahasiswa secara aman serta membatasi akses sesuai hak pengguna (admin, dosen, atau staf akademik).

2.4 Kebutuhan Non-fungsional

Kebutuhan nonfungsional manajemen data mahasiswa adalah persyaratan yang berkaitan dengan kualitas, kinerja, dan batasan sistem, bukan pada fungsi utama sistem. Kebutuhan ini memastikan sistem berjalan dengan baik, aman, dan nyaman digunakan. Berikut penjelasannya:

2.4.1 Keamanan (*Security*)

Sistem harus melindungi data mahasiswa dari akses tidak sah dengan mekanisme login, enkripsi data, dan pengaturan hak akses pengguna.

2.4.2 Kinerja (*Performance*)

Sistem mampu memproses dan menampilkan data dengan cepat, meskipun digunakan oleh banyak pengguna secara bersamaan.

2.4.3 Keandalan (*Reliability*)

Sistem dapat berjalan secara stabil dan meminimalkan kesalahan atau kegagalan saat pengelolaan data.

2.4.4 Ketersediaan (*Availability*)

Sistem dapat diakses kapan saja sesuai kebutuhan akademik, dengan waktu henti (downtime) seminimal mungkin.

2.4.5 Kemudahan Penggunaan (*Usability*)

Antarmuka sistem mudah dipahami dan digunakan oleh admin, dosen, maupun staf akademik.

2.4.6 Skalabilitas (*Scalability*)

Sistem mampu menangani pertambahan jumlah data mahasiswa dan pengguna tanpa penurunan kinerja.

2.4.7 Pemeliharaan (*Maintainability*)

Sistem mudah diperbaiki, diperbarui, dan dikembangkan apabila terdapat perubahan kebutuhan.

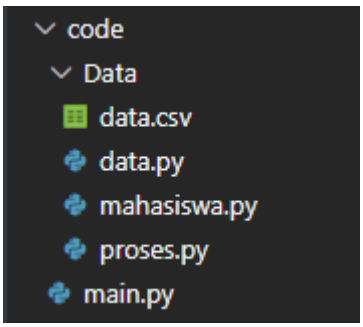
2.4.8 Portabilitas (*Portability*)

Sistem dapat dijalankan pada berbagai perangkat atau sistem operasi jika diperlukan.

2.5 Penjelasan Struktur Program

Program ini dibangun menggunakan **pendekatan modular**, di mana setiap file memiliki tanggung jawab yang jelas. Struktur program bertujuan agar kode mudah dipahami, dirawat, dan dikembangkan.

Struktur Folder



Gambar 2. 1

Penjelasan Tiap File

2.5.1 Menu.py

Berfungsi sebagai **pengendali utama aplikasi (main program)**. File ini:

1. Menampilkan *menu*
2. Mengatur alur navigasi program

3. Memanggil fungsi dari proses.py

2.5.2 Mahasiswa.py

Berisi **class Mahasiswa** yang merepresentasikan satu objek mahasiswa, meliputi:

1. NIM
2. Nama
3. Nilai akademik
4. Data kehadiran

Selain itu, file ini juga berisi:

1. NilaiAkhir() → menghitung nilai akhir mahasiswa
2. *Grade*() → menentukan *grade* berdasarkan nilai akhir

2.5.3 Data.py

Berfungsi sebagai **pengelola penyimpanan data**, meliputi:

1. Membaca data dari file CSV
2. Menyimpan data ke file CSV
3. Menghasilkan NIM otomatis

File ini menjadi penghubung antara **program** dan **penyimpanan data**.

2.5.4 Proses.py

Berisi **logika utama aplikasi**, seperti:

1. CRUD data mahasiswa
2. *Input* dan *update* nilai akademik
3. Presensi mahasiswa
4. Pembuatan laporan berbentuk tabel

2.6 Penjelasan Logika Khusus

Program ini memiliki beberapa logika khusus yang penting.

2.6.1 Presensi Maksimal 16 Pertemuan

1. Setiap mata kuliah dibatasi maksimal **16 pertemuan**
2. Data presensi disimpan dalam bentuk ***dictionary***

Keterangan:

1. H → Hadir
2. A → Alpha
3. I → Izin
4. - → Tidak ada data

Jika mahasiswa belum ada saat pertemuan sebelumnya, otomatis diisi strip (-).

2.6.2 Validasi Input

Program menerapkan validasi ketat:

1. Nilai akademik hanya boleh **1–100**
2. Status kehadiran hanya menerima **H, A, I, -**
3. Pertemuan hanya **1–16**

Jika *input* tidak valid, program akan meminta *input* ulang.

2.6.3 Advanced *Function* dan *Grade*

Lambda function digunakan untuk menghitung nilai akhir mahasiswa berdasarkan nilai tugas, UTS, dan UAS.

Lambda dipilih karena perhitungannya bersifat sederhana dan tidak memerlukan fungsi terpisah, sehingga kode menjadi lebih ringkas dan mudah dibaca.

```
def nilaiAkhir(self):
    nilaiAkhir = lambda tugas, uts, uas: (0.30 * tugas) + (0.35 * uts) + (0.35 * uas)
    return nilaiAkhir(self.nilaiTugas, self.nilaiUts, self.nilaiUas)
```

Gambar 2. 2

Recursive function digunakan pada sistem menu program. Setelah suatu proses dijalankan, fungsi menu akan memanggil dirinya kembali untuk menampilkan menu berikutnya hingga pengguna memilih keluar.

Pendekatan ini digunakan untuk menghindari penggunaan perulangan tanpa batas (*While True*) serta memberikan kontrol alur program yang lebih jelas dan terstruktur.

```
def menuAwal():
    print("\n<========== PROJECT ==========>")
    print("1. Manajemen Data Mahasiswa")
    print("2. Nilai Akademik")
    print("3. Presensi")
    print("4. Laporan")
    print("5. Exit")

    try:
        pil = int(input("Masukan Pilihan : ").strip())
    except ValueError:
        print("Input harus angka!")
        return menuAwal() # Itulah 4 trio

    if pil == 1:
        return menuManajemen()
    elif pil == 2:
        proses.inputNilaiMhs(dataCsvMhs)
    elif pil == 3:
        proses.presensiMhs(dataCsvMhs)
    elif pil == 4:
        proses.laporanMhs(dataCsvMhs)
    elif pil == 5:
        print("Program Selesai...")
        return
    else:
        print("Pilihan tidak tersedia!")

    return menuAwal() # Itulah 4 trio
```

Gambar 2. 3

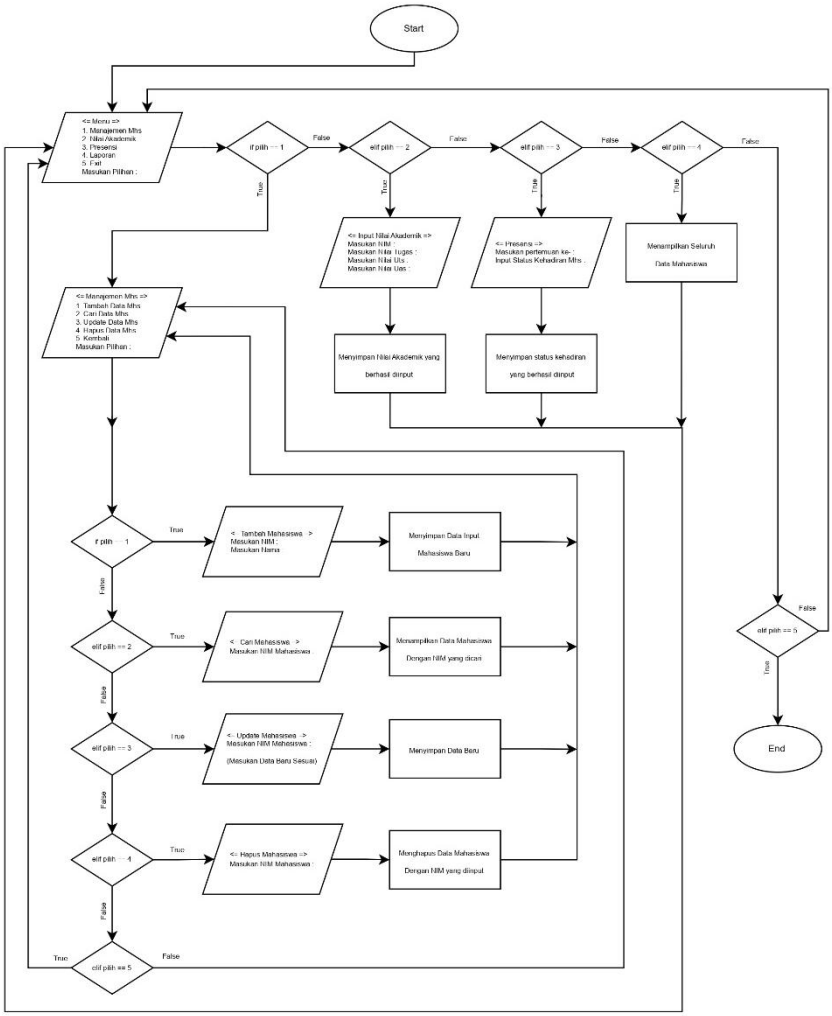
Grade ditentukan berdasarkan nilai akhir:

1. A ≥ 80
2. B ≥ 70
3. C ≥ 55
4. D ≥ 40
5. E < 40

2.6.4 Pembuatan Tabel Dinamis

Lebar kolom **nama** menyesuaikan panjang nama terpanjang secara otomatis agar tampilan tetap rapi.

2.7 *Flowchart* (Diagram Alur)

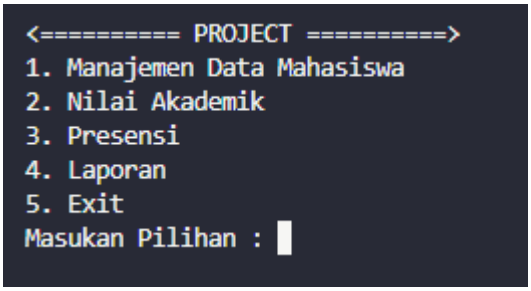


Gambar 2. 4

2.8 Hasil *Output*

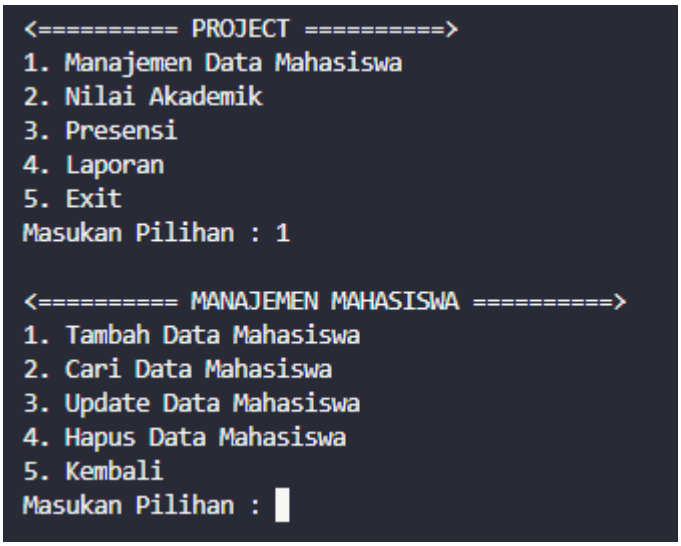
Berikut merupakan hasil output dari program berikut ini :

Pertama kita run dulu projectnya di terminal menggunakan *py main.py* maka tampilah *interface* pertama kita yaitu menu awal pada program ini!



Gambar 2. 5

Selanjutnya kita akan mencoba fitur **Manajemen Data Mahasiswa** dengan menginputkan angka 1 lalu tekan *enter*! Maka tampilah menu *interface* manajemen mahasiswa!



Gambar 2. 6

Selanjutnya kita akan mencoba fitur **Tambah Data Mahasiswa** dengan menginputkan angka 1 lalu tekan *enter*!

```
<========== MANAJEMEN MAHASISWA ==========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan : 1
NIM Mahasiswa baru : 0001
Masukan Nama : Ryantha
Data Mahasiswa Berhasil Ditambahkan!

<========== MANAJEMEN MAHASISWA ==========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan :
```

Gambar 2. 7

Setelah kita menginputkan angka 1 maka program akan menampilkan **NIM Mahasiswa baru *otomatis*** dan *users* disuruh menginputkan nama baru!

Mengapa NIM-nya bisa otomatis? Karena kita sudah membuat sebuah fungsi *nimOtomatis()* dimana jika belum ada data mahasiswa maka sistem akan menambahkan NIM 0001 secara otomatis menggunakan fungsi ini :

```python
def nimOtomatis():
    with open("Data/data.csv", "r", newline="") as file:
        ngambilNim = csv.reader(file)
        next(ngambilNim)

        Nim = []

        for i in ngambilNim:
            if i:
                Nim.append(int(i[0]))

        if not Nim:
            return "0001"

        nimBesar = max(Nim) + 1
        return str(nimBesar).zfill(4)
```

Gambar 2. 8

Lalu bagaimana jika data mahasiswanya sudah ada? Maka sistem akan membaca NIM data mahasiswa yang *terbesar* lalu ditambahkan dengan angka 1!

Misal terdapat sebuah mahasiswa dengan NIM 0001, lalu user ingin menambahkan mahasiswa baru lagi, otomatis sistem akan membaca NIM sebelumnya dan ditambahkan dengan angka 1 menjadi 0002!

```
<========== MANAJEMEN MAHASISWA ==========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan : 1
NIM Mahasiswa baru : 0002
Masukan Nama : Alice
Data Mahasiswa Berhasil Ditambahkan!
```

Gambar 2. 9

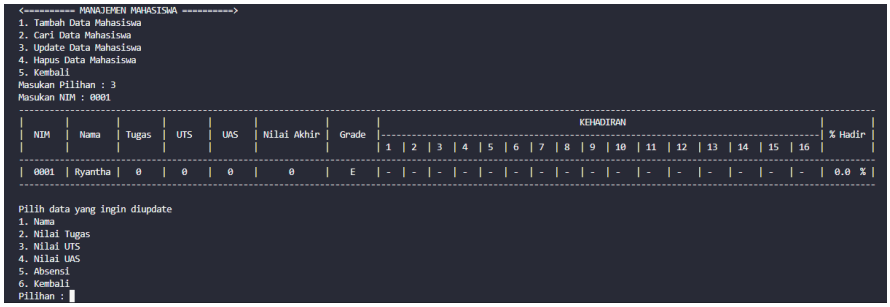
Oke selanjutnya kita akan mencoba fitur **Cari Data Mahasiswa** dengan menginputkan angka 2 lalu tekan *enter* :

```
<========= MANAJEMEN MAHASISWA =========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan : 2
Cari Mahasiswa dengan NIM : 0001
-------------------------------------------------------------------------------------------------------------------------
|      |        |       |     |     |           |       |                                                       |          |
| NIM  | Nama   | Tugas | UTS | UAS | Nilai Akhir | Grade |                       KEHADIRAN                       | % Hadir  |
|      |        |       |     |     |           |       | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |          |
-------------------------------------------------------------------------------------------------------------------------
| 0001 | Ryantha |  0  |  0  |  0  |     0     |   E   | - | - | - | - | - | - | - | - | - | -  | -  | -  | -  | -  | -  | -  |  0.0 %   |
-------------------------------------------------------------------------------------------------------------------------
```

Gambar 2. 10

Hasilnya sistem akan meminta user menginputkan NIM mahasiswa yang ingin dicari lalu menampilkan seluruh data yang ada pada mahasiswa tersebut!

Selanjutnya kita akan mencoba fitur **Update Data Mahasiswa** dengan menginputkan angka 3 lalu *enter*!



Gambar 2. 11

Disini *user* disuruh menginputkan NIM mahasiswa yang datanya ingin diubah, lalu tampil data mahasiswa tersebut dan terdapat menu untuk menginputkan data yang ingin dirubah, seperti nama, nilai tugas, nilai uts, dan absensi.

Lalu mari kita coba satu-satu semuanya!

1. Ubah Nama

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 1
Input Nama baru : Riantha
```

Gambar 2. 12

2. Ubah Nilai Tugas

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 2
Input Nilai Tugas baru : 95
```

Gambar 2. 13

3. Ubah Nilai Uts

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 3
Input Nilai Uts baru : 95
```

Gambar 2. 14

4. Ubah Nilai UAS

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 4
Input Nilai Uas baru : 95
```

Gambar 2. 15

5. Ubah Absensi Pertemuan Ke-n

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 5
Update absensi pertemuan ke- (1-16) : 1
Status lama : -
Status baru (H/A/I/-) : h
Data Mahasiswa dengan NIM 0001 Berhasil Diperbaharui!!
```

Gambar 2. 16

Untuk Fitur **Kembali** pada fitur **Update Data Mahasiswa** itu akan kembali ke menu Manajemen Data Mahasiswa :

```
Pilih data yang ingin diupdate
1. Nama
2. Nilai Tugas
3. Nilai UTS
4. Nilai UAS
5. Absensi
6. Kembali
Pilihan : 6

<========== MANAJEMEN MAHASISWA ==========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan :
```

Gambar 2. 17

Sedangkan untuk fitur **Kembali** pada Manajemen Mahasiswa itu akan mengarah ke menu awal!

Fungsi **Hapus Data Mahasiswa** :

```
<========== MANAJEMEN MAHASISWA ==========>
1. Tambah Data Mahasiswa
2. Cari Data Mahasiswa
3. Update Data Mahasiswa
4. Hapus Data Mahasiswa
5. Kembali
Masukan Pilihan : 4
Masukan NIM mahasiswa yang ingin dihapus : 0002
Mahasiswa dengan NIM : 0002 berhasil dihapus!!
```

Gambar 2. 18

Fungsi **Nilai Akademik** :

```
<========== PROJECT ==========>
1. Manajemen Data Mahasiswa
2. Nilai Akademik
3. Presensi
4. Laporan
5. Exit
Masukan Pilihan : 2
Masukan NIM : 0001
Mahasiswa dengan NIM 0001 adalah Riantha

Masukan nilai Tugas (1-100) : 99
Masukan nilai UTS   (1-100) : 99
Masukan nilai UAS   (1-100) : 99
Nilai berhasil diinput!
```

Gambar 2. 19

Fungsi **Presensi** :

```
<========== PROJECT ==========>
1. Manajemen Data Mahasiswa
2. Nilai Akademik
3. Presensi
4. Laporan
5. Exit
Masukan Pilihan : 3
Masukan pertemuan ke- (1-16) : 2

<= PERTEMUAN 2 (PRAKTIKUM) =>

Riantha (H/A/I/-): h

Presensi pertemuan berhasil disimpan.
```

Gambar 2. 20

Fungsi **Laporan** :

Gambar 2. 21

BAB III
PENUTUP

3.1 Kesimpulan

Sistem manajemen data mahasiswa ini telah memenuhi kebutuhan dasar sistem informasi akademik, baik dari sisi konsep maupun implementasi. Data dikelola secara sistematis, konsisten, dan dapat diolah menjadi informasi yang berguna, menunjukkan pemahaman yang baik terhadap tujuan sistem.

Fungsi utama seperti *CRUD*, pencarian, validasi, serta dukungan kebutuhan non-fungsional dasar telah terpenuhi dan sesuai untuk skala akademik sederhana. Struktur program yang modular mencerminkan pola berpikir terorganisir dan menjadi fondasi yang mudah dikembangkan. Penerapan logika khusus—seperti pembatasan presensi, validasi *input*, perhitungan nilai, dan tampilan tabel dinamis—menunjukkan bahwa sistem dirancang berdasarkan aturan nyata, bukan sekadar simulasi.

3.2 Saran

Ke depan, sistem ini dapat ditingkatkan dengan mengganti penyimpanan berbasis CSV menjadi basis data relasional agar data lebih terjamin keamanannya dan pengelolaannya lebih efisien. Selain itu, perlu ditambahkan mekanisme login serta pengaturan hak akses pengguna untuk membedakan peran admin, dosen, dan staf akademik.

Sistem juga sebaiknya dilengkapi dengan penanganan kesalahan, pencatatan aktivitas, dan pengujian yang lebih menyeluruh agar lebih stabil dan mudah dikembangkan. Penggunaan antarmuka berbasis *GUI* atau *web* akan

membuat sistem lebih praktis, nyaman digunakan, dan terlihat lebih profesional.

DAFTAR PUSTAKA

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database system concepts* (7th ed.). McGraw-Hill Education.

Connolly, T., & Begg, C. (2015). *Database systems: A practical approach to design, implementation, and management* (6th ed.). Pearson Education.

Pressman, R. S., & Maxim, B. R. (2015). *Software engineering: A practitioner's approach* (8th ed.). McGraw-Hill Education.

Connolly, T., & Begg, C. (2015). *Database systems: A practical approach to design, implementation, and management* (6th ed.). Pearson Education.

IEEE. (2018). *ISO/IEC/IEEE 29148:2018 systems and software engineering — Life cycle processes — Requirements engineering*. IEEE Standards Association.