

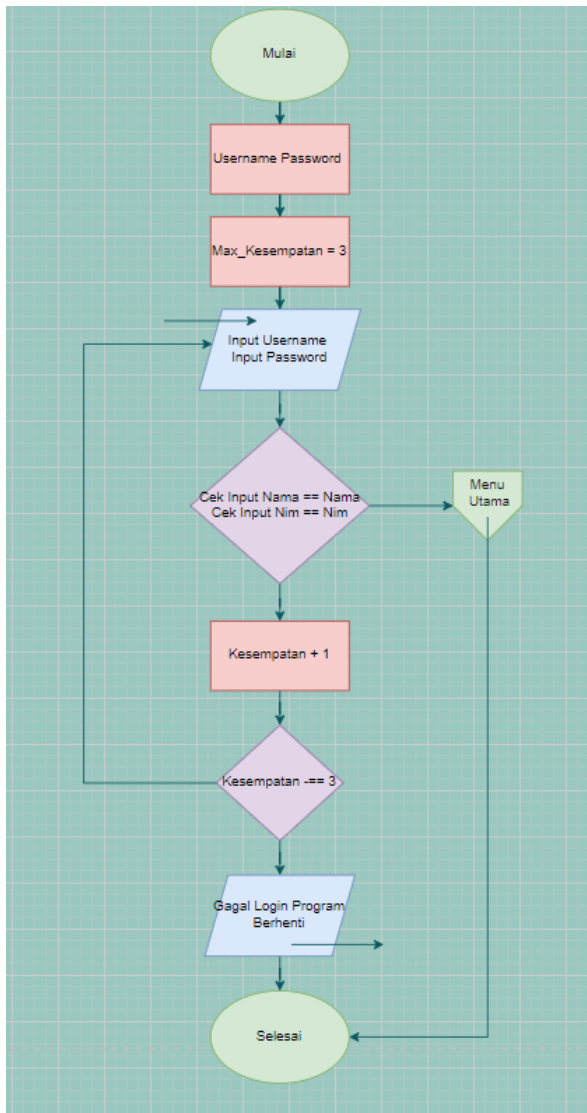
LAPORAN PRAKTIKUM
POSTTEST 2
ALGORITMA PEMROGRAMAN
LANJUT



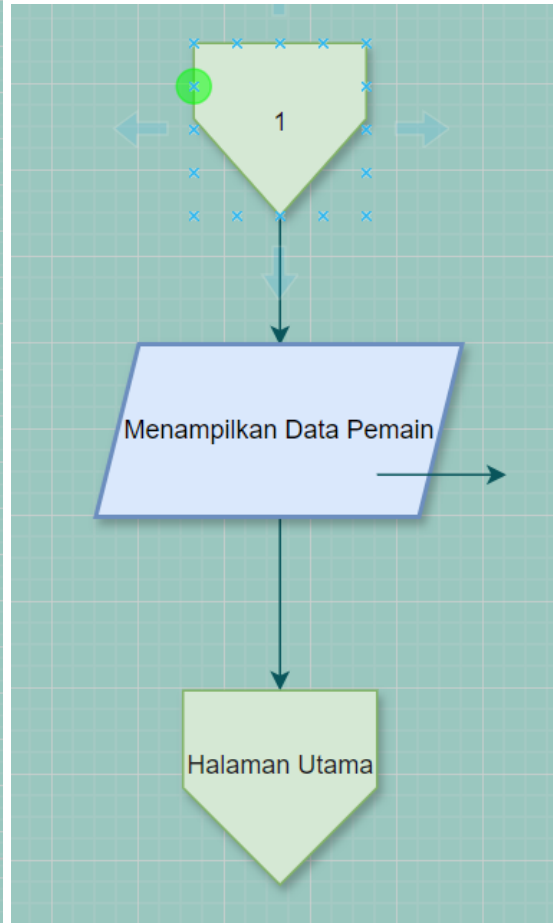
Disusun oleh:
Rusdiana (2409106021)
Kelas A1'24

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

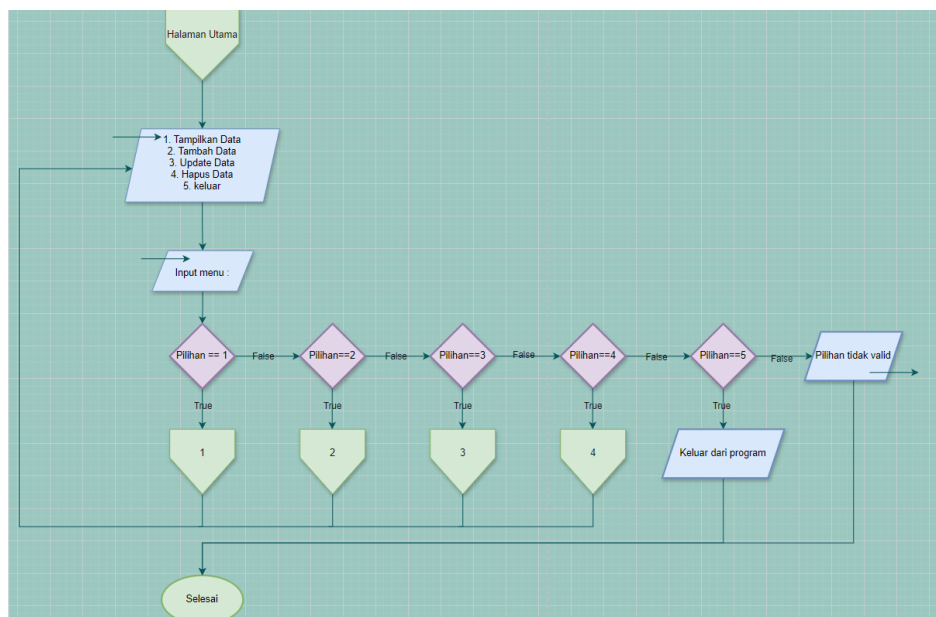
1. Flowchart



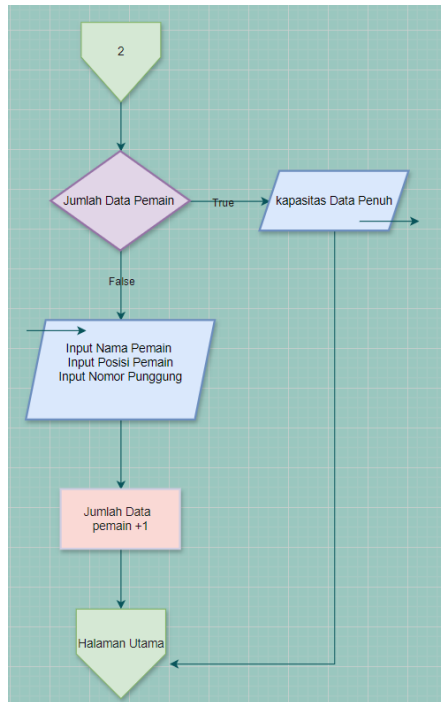
Gambar 1.1 Login



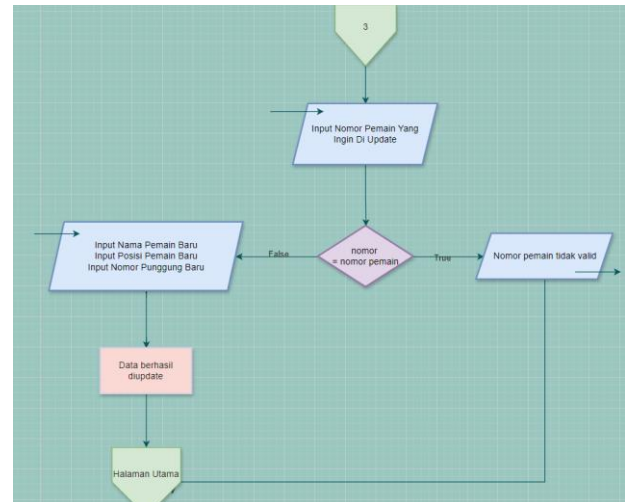
Gambar 1.3 Menampilkan data



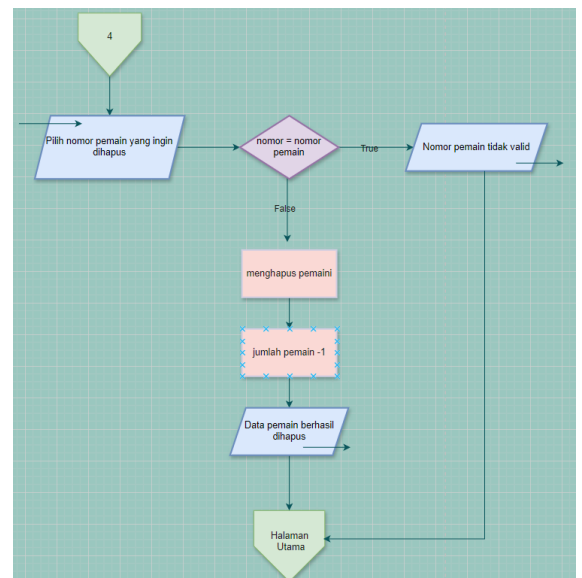
Gambar 1.2 Halaman Utama



Gambar 1. 4 Penambahan Data



Gambar 1.5 Update Pemain



Gambar 1. 6 Penghapusan Data

2. Analisis Program

2.1 Deskripsi Singkat Program

Program ini adalah sebuah manajemen “Data Pemain Bola” yang ditulis dalam bahasa C++. Program ini memungkinkan pengguna untuk melakukan berbagai operasi CRUD (Create, Read, Update, Delete) pada data pemain bola, seperti menambahkan, menampilkan, mengupdate, dan menghapus data pemain. Selain itu, program ini juga dilengkapi dengan sistem login untuk mengamankan akses ke aplikasi..

2.2 Penjelasan Alur & Algoritma

Program ini terdiri dari beberapa tahap utama, yang mencakup proses login, navigasi menu, serta operasi CRUD (Create, Read, Update, Delete) pada data produk. Berikut adalah alur kerja program :

1. Login Pengguna

- Pengguna harus memasukkan username dan 3 digit terakhir NIM untuk masuk ke sistem.
- Jika pengguna salah memasukkan data login sebanyak tiga kali, program akan berhenti.
- Jika login berhasil, pengguna diarahkan ke menu utama.

2. Menampilkan Menu Utama

- Program menampilkan opsi yang bisa dipilih oleh pengguna:
 1. Tampilkan Data Pemain
 2. Tambah Data Pemain
 3. Update Data Pemain
 4. Hapus Data Pemain
 5. Keluar
- Pengguna memasukkan nomor pilihan menu, yang akan menentukan langkah selanjutnya.

3. CRUD Data Produk

- **Tampilan Data Pemain:** Program mencetak daftar data pemain dalam format tabel agar lebih rapi dan mudah dibaca.
- **Tambah Data Pemain:** Pengguna menginputkan nama pemain, posisi pemain, dan nomor punggung pemain, yang kemudian disimpan dalam array dua dimensi.

- **Update Data Pemain:** Pengguna memilih Pemain berdasarkan nomor urutnya, lalu memasukkan data baru untuk mengganti informasi yang ada.
- **Hapus Data Pemain :** Pengguna memilih nomor pemain yang ingin dihapus, lalu program menggeser data dalam array agar tetap terstruktur.

4. Keluar dari Program

- Jika pengguna memilih opsi Keluar, program akan menampilkan pesan perpisahan dan berhenti.

Penjelasan Fungsi dalam Program:

- **TampilkanData()** Menampilkan daftar pilihan menu utama.
- **TambahData()** Menambahkan data pemain baru ke dalam array jika kapasitas belum penuh.
- **TampilkanData()** Menampilkan semua data pemain yang telah disimpan dalam format tabel.
- **UbahData()** Mengubah informasi data pemain tertentu berdasarkan input pengguna.
- **HapusData()** Menghapus data pemain yang dipilih dan menyesuaikan susunan array.
- **Main()** Mengontrol seluruh proses eksekusi program dari login hingga interaksi dengan menu.

3. Source Code

A. Login

Pada Fitur ini menu yang digunakan untuk memvalidasi user yang ingin menggunakan aplikasi

Source Code :

```
int main() {
    // Inisialisasi variabel array satu dimensi
    string nama_pemain[100];
    string posisi_pemain[100];
    int nomor_punggung[100];
    int jumlah_data = 0;

    const string username = "Rusdiana";
    const string password = "2409106021";
    const int max_kesempatan = 3;
    int kesempatan = 0;

    while (kesempatan < max_kesempatan) {
        string input_username, input_password;
        cout << "Username: ";
        getline(cin, input_username);
        cout << "Password: ";
        getline(cin, input_password);

        if (input_username == username && input_password == password) {
            cout << "Login berhasil!\n" << endl;
            break;
        } else {
            kesempatan++;
            cout << "Login gagal. Sisa percobaan: " << max_kesempatan -
kesempatan << "\n" << endl;
        }
    }

    if (kesempatan == max_kesempatan) {
        cout << "Anda telah mencoba login " << max_kesempatan << " kali. Program
berhenti." << endl;
        return 0;
    }
}
```

B. Menu

Terdapat beberapa menu untuk memilih fitur pada aplikasi

Source Code

```
// Main program
while (true) {
    cout << "\n==== Manajemen Data Pemain Bola =====> endl;
    cout << "1. Tampilkan Data" << endl;
    cout << "2. Tambah Data" << endl;
    cout << "3. Update Data" << endl;
    cout << "4. Hapus Data" << endl;
    cout << "5. Keluar" << endl;
    int pilihan;
    cout << "Pilih menu (1-5): ";
    cin >> pilihan;
```

C. CRUD Manajemen Data Pemain Bola

Fitur CRUD dalam pemrograman ini mengacu pada empat operasi utama yaitu Create (membuat/menambahkan data baru), Read (Mengambil dan menampilkan data yang tersimpan), Update (Mengubah atau memperbarui data yang sudah ada.), Delet (Menghapus data dari Array.) yang digunakan untuk mengelola data produk dalam Array

Source Code:

```
if (pilihan == 1) {
    // Tampilkan Data
    cout << "\n===== Data Pemain Bola =====> endl;
    cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) <<
    "Posisi" << setw(20) << "Nomor Punggung" << endl;
    cout << string(66, '-') << endl;
    for (int i = 0; i < jumlah_data; i++) {
        cout << setw(5) << i + 1 << setw(20) << nama_pemain[i] << setw(20) <<
        posisi_pemain[i] << setw(15) << nomor_punggung[i] << endl;
    }
    cout << endl;
} else if (pilihan == 2) {
    // Tambah Data
    if (jumlah_data < 100) {
        cout << "Masukkan Nama Pemain: ";
        cin.ignore();
        getline(cin, nama_pemain[jumlah_data]);
        cout << "Masukkan Posisi Pemain: ";
        getline(cin, posisi_pemain[jumlah_data]);
        cout << "Masukkan Nomor Punggung: ";
        cin >> nomor_punggung[jumlah_data];
        jumlah_data++;
        cout << "Data berhasil ditambahkan!\n";
    } else {
        cout << "Kapasitas data penuh!\n";
    }
}
```



```

    } else if (pilihan == 3) {
        // Update Data
        if (jumlah_data > 0) {
            cout << "\n==== Data Pemain Bola =====> << endl;
            cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) <<
"Posisi" << setw(15) << "Nomor Punggung" << endl;
            cout << string(60, '-') << endl;
            for (int i = 0; i < jumlah_data; i++) {
                cout << setw(5) << i + 1 << setw(20) << nama_pemain[i] <<
setw(20) << posisi_pemain[i] << setw(15) << nomor_punggung[i] << endl;
            }
            cout << endl;

            int index;
            cout << "Masukkan nomor pemain yang ingin diupdate: ";
            cin >> index;
            if (index >= 1 && index <= jumlah_data) {
                cout << "Masukkan Nama Pemain Baru: ";
                cin >> nama_pemain[index - 1];
                cout << "Masukkan Posisi Pemain Baru: ";
                cin >> posisi_pemain[index - 1];
                cout << "Masukkan Nomor Punggung Baru: ";
                cin >> nomor_punggung[index - 1];
                cout << "Data berhasil diupdate!\n";
            } else {
                cout << "Nomor pemain tidak valid!\n";
            }
        } else {
            cout << "Tidak ada data untuk diupdate!\n";
        }
    } else if (pilihan == 4) {
        // Hapus Data
        if (jumlah_data > 0) {
            cout << "\n=== Data Pemain Bola ===> << endl;
            cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) <<
"Posisi" << setw(15) << "Nomor Punggung" << endl;
            cout << string(60, '-') << endl;
            for (int i = 0; i < jumlah_data; i++) {
                cout << setw(5) << i + 1 << setw(20) << nama_pemain[i] <<
setw(20) << posisi_pemain[i] << setw(15) << nomor_punggung[i] << endl;
            }
            cout << endl;

            int index;
            cout << "Masukkan nomor pemain yang ingin dihapus: ";
            cin >> index;
            if (index >= 1 && index <= jumlah_data) {
                for (int i = index - 1; i < jumlah_data - 1; i++) {
                    nama_pemain[i] = nama_pemain[i + 1];
                    posisi_pemain[i] = posisi_pemain[i + 1];
                }
            }
        }
    }
}

```

```
        nomor_punggung[i] = nomor_punggung[i + 1];
    }
    jumlah_data--;
    cout << "Data berhasil dihapus!\n";
} else {
    cout << "Nomor pemain tidak valid!\n";
}
} else {
    cout << "Tidak ada data untuk dihapus!\n";
}
```

4. Uji Coba dan Hasil Output

4.1 Uji Coba

Untuk memastikan bahwa program berjalan dengan baik dilakukan beberapa skenario uji coba dengan berbagai jenis input. Berikut adalah dua skenario utama yang diuji:

1. Skenario 1: Login dengan Input yang Salah

- **Langkah-langkah:**

1. Menjalankan program.
2. Memasukkan username atau NIM yang salah sebanyak tiga kali.
3. Program menampilkan pesan error dan berhenti setelah tiga kali percobaan gagal.

- Hasil yang diharapkan:

- Jika pengguna salah memasukkan login tiga kali, program akan berhenti secara otomatis.
- Jika pengguna memasukkan data yang benar sebelum tiga kali percobaan, mereka dapat mengakses menu utama.

2. Skenario 2: Menambahkan dan Menampilkan Data Pemain

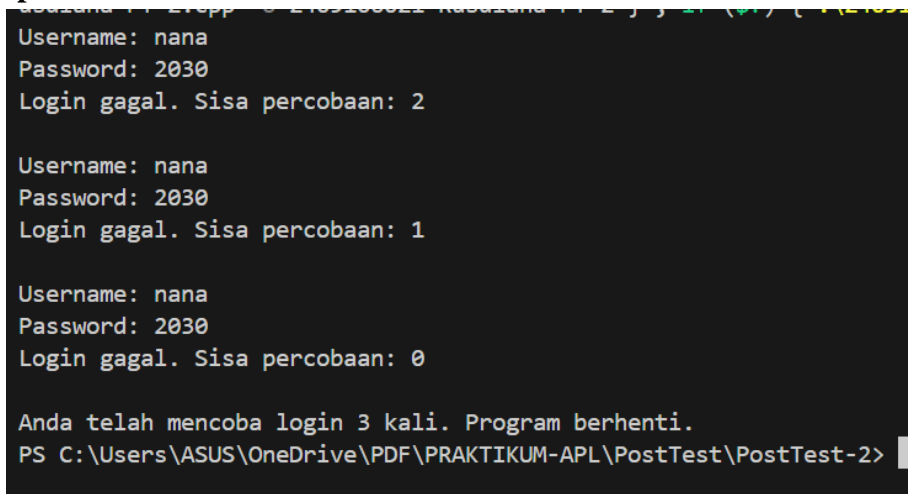
- Langkah-langkah:

1. Login dengan username dan NIM yang benar.
2. Memilih menu "Tampilkan Data" dan memasukkan data pemain (nama, posisi, nomor punggung).
3. Memilih menu "Tambah Data" untuk memastikan data yang telah dimasukkan muncul dengan format tabel yang rapi.

- Hasil yang diharapkan:

- Data yang baru ditambahkan muncul dalam daftar dengan format tabel.
- Data pemain tersimpan dengan benar dalam array.

4.2 Hasil Output



```
Username: nana
Password: 2030
Login gagal. Sisa percobaan: 2

Username: nana
Password: 2030
Login gagal. Sisa percobaan: 1

Username: nana
Password: 2030
Login gagal. Sisa percobaan: 0

Anda telah mencoba login 3 kali. Program berhenti.
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL\PostTest\PostTest-2>
```

Gambar 4. 2 Gagal Login

```
Username: Rusdiana
Password: 2409106021
Login berhasil!

==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): █
```

Gambar 4. 1 Berhasil Login

```
==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 2
Masukkan Nama Pemain: Neymar Jr
Masukkan Posisi Pemain: Gelandang
Masukkan Nomor Punggung: 11
Data berhasil ditambahkan!
```

Gambar 4. 4 Tambah Data

```
==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 1

===== Data Pemain Bola =====
No          Nama Pemain          Posisi          Nomor Punggung
-----
1           Neymar Jr            Gelandang       11
```

Gambar 4. 3 Tampilkan Data Pemain

```

==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 3

===== Data Pemain Bola =====
      No          Nama Pemain          Posisi Nomor Punggung
-----
      1           Neymar Jr           Gelandang           11

Masukkan nomor pemain yang ingin diupdate: 1
Masukkan Nama Pemain Baru: Messi
Masukkan Posisi Pemain Baru: Keeper
Masukkan Nomor Punggung Baru: 10
Data berhasil diupdate!

```

Gambar 4. 6 Update Data Pemain

```

Data berhasil diupdate!

==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 1

===== Data Pemain Bola =====
      No          Nama Pemain          Posisi  Nomor Punggung
-----
      1           Messi               Keeper   10

```

Gambar 4. 5 daftar setelah diubah

```

==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 4

=== Data Pemain Bola ===
      No          Nama Pemain          Posisi Nomor Punggung
-----
      1           Messi               Keeper   10
      2           Yamal               Stc      12

```

Gambar 4. 8 Hapus Data Pemain

```
==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 1

===== Data Pemain Bola =====
No          Nama Pemain          Posisi          Nomor Punggung
-----
1           Yamal                      Stc              12
```

Gambar 4. 7 Daftar Setelah Dihapus

```
==== Manajemen Data Pemain Bola ====
1. Tampilkan Data
2. Tambah Data
3. Update Data
4. Hapus Data
5. Keluar
Pilih menu (1-5): 5
Terima kasih telah menggunakan program ini!
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL\PostTest\PostTest-2>
```

Gambar 4. 9 Keluar Dari Program

5. Langkah-Langkah GIT

1. Git Init

Git init adalah perintah Git yang digunakan untuk menginisialisasi repository Git baru dalam sebuah folder. Cukup ketik “git init” pada terminal vscode

```
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL> git init
Reinitialized existing Git repository in C:/Users/ASUS/OneDrive/PDF/PRAKTIKUM-APL/.git/
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL>
```

Gambar 5. 1 git_init

2. Git add

Git add adalah perintah Git yang digunakan untuk menambahkan perubahan pada file ke staging area sebelum dilakukan commit. Perintah ini memungkinkan Git mengetahui file mana saja yang akan dimasukkan dalam commit berikutnya

```
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL> git add .
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL>
```

Gambar 5. 2 Git_add

3. Git Commit

Git commit adalah perintah Git yang digunakan untuk menyimpan perubahan yang telah ditambahkan ke staging area ke dalam repository. Setiap commit akan memiliki hash unik, pesan commit, dan menyimpan snapshot dari perubahan yang dilakukan.

```
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL> git commit -m "finish post test 2"
[main e648277] finish post test 2
 9 files changed, 327 insertions(+)
 create mode 100644 .vscode/tasks.json
 create mode 100644 Kelas/pertemuan_2/pertemuan-2.cpp
 create mode 100644 Kelas/pertemuan_2/pertemuan-2.exe
 delete mode 100644 PostTest/PostTest-1/2409106021-RUSDIAI-PT-1.docx
 create mode 100644 PostTest/PostTest-1/2409106021-Rusdiana-PT-1 .pdf
 create mode 100644 PostTest/PostTest-1/Untitled-1.cpp
 create mode 100644 PostTest/PostTest-2/2409106021-Rusdiana-PT-2.cpp
 create mode 100644 PostTest/PostTest-2/2409106021-Rusdiana-PT-2.docx
 create mode 100644 PostTest/PostTest-2/2409106021-Rusdiana-PT-2.exe
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL>
```

Gambar 5. 3 Git_Commit

4. Git Push

Git push adalah perintah Git yang digunakan untuk mengirim (mengunggah) perubahan dari repository lokal ke repository remote (seperti GitHub, GitLab, atau Bitbucket). Perintah ini memastikan bahwa perubahan yang sudah dikomit di lokal tersedia di repository jarak jauh sehingga bisa diakses oleh orang lain atau untuk cadangan.

```
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL> git push origin main --force
>>
Enumerating objects: 21, done.
Counting objects: 100% (21/21), done.
Delta compression using up to 8 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (21/21), 5.84 MiB | 2.41 MiB/s, done.
Total 21 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
remote: This repository moved. Please use the new location:
remote:   https://github.com/RusdianaNana/praktikum-apl.git
To https://github.com/RusdianaNana/Praktikum-Apl.git
+ 0d25b4e...e648277 main -> main (forced update)
PS C:\Users\ASUS\OneDrive\PDF\PRAKTIKUM-APL>
```

Gambar 5. 5 Git_Push