**LAPORAN PRAKTIKUM POSTTEST 3**

**ALGORITMA PEMROGRAMAN LANJUT**



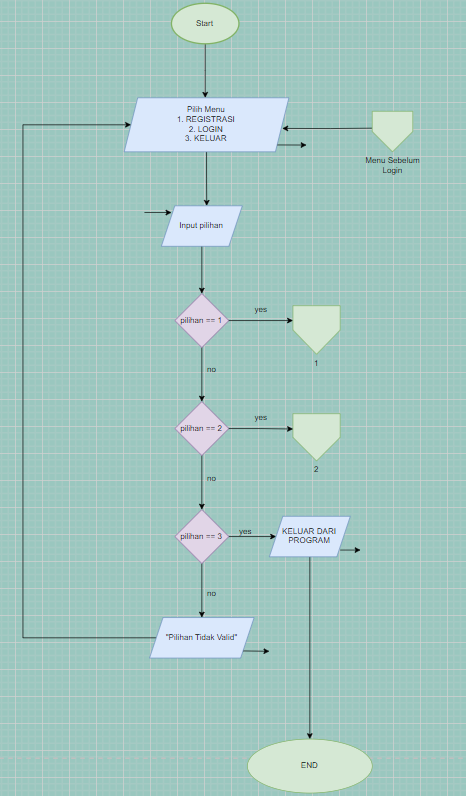
# Disusun oleh: Rusdiana (2409106021)

**Kelas A1’24**

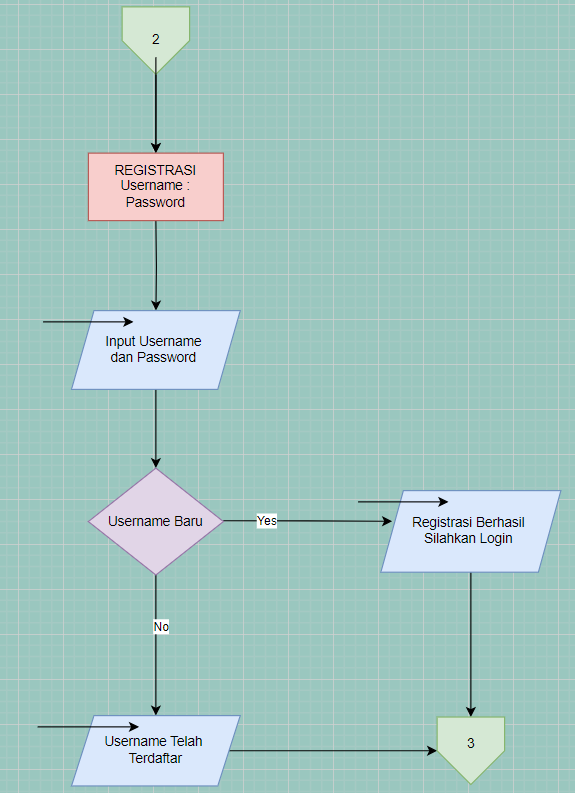
**PROGRAM STUDI INFORMATIKA UNIVERSITAS MULAWARMAN SAMARINDA**

**2025**

# Flowchart

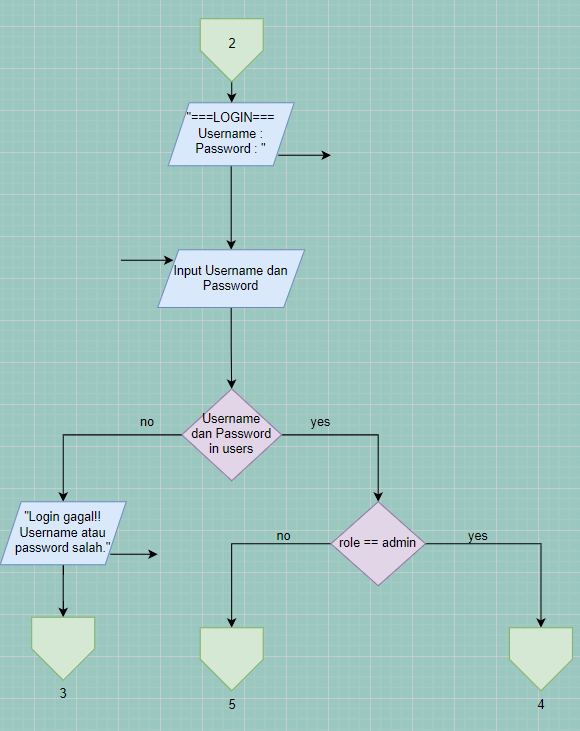
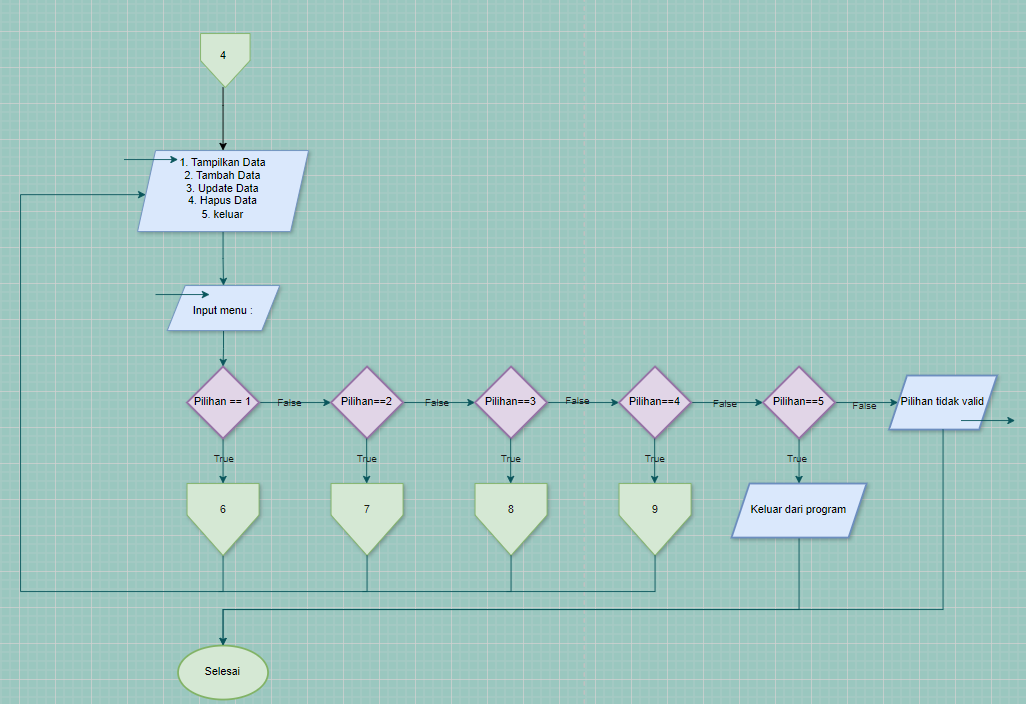


Gambar 1.1 Menu Sebelum Login

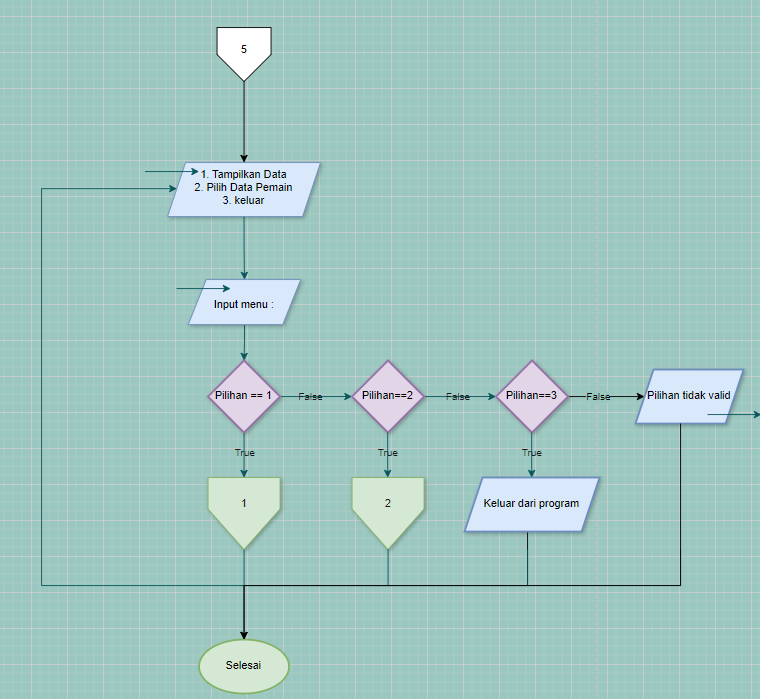


**j**

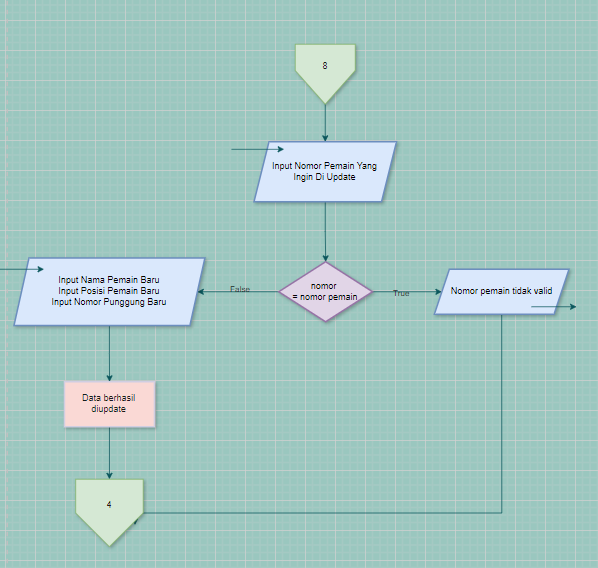
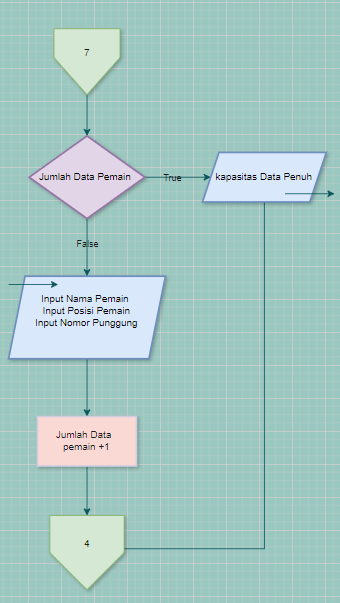
Gambar 1.2 Menu Registrasi

Gambar 1.4 Menu Admin 

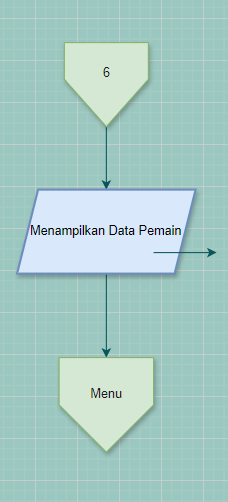
Gambar 1.3 Menu Login



Gambar 1.5 Menu User

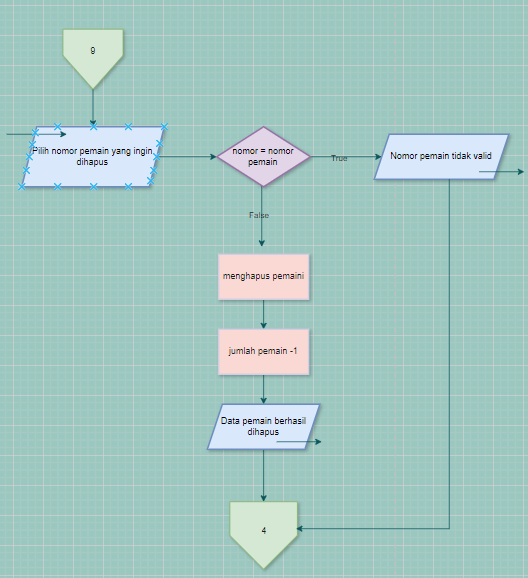
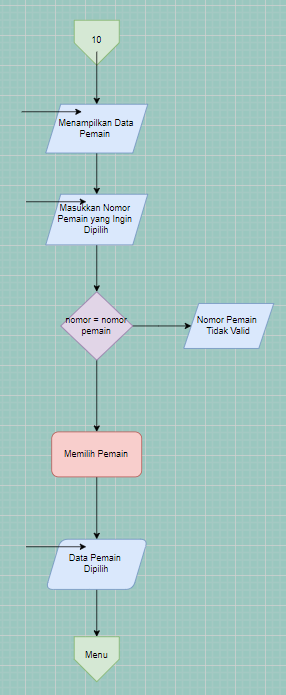


Gambar 1.7 Tambah Data



Gambar 1.6 Tampilan Data

Gambar 1.8 Update Data

Gambar 1.9 Hapus Data 

Gambar 1.10 Pilih Data Pemain

### Analisis Program

* 1. **Deskripsi Singkat Program**

Program ini adalah sebuah manajemen “Data Pemain Bola” yang ditulis dalam bahasa C++. Pada program ini terdapat menu sebagai user baru dan menu sebagai admin, di mana menu user sendiri dapat mengakses tampilan, pilihan, serta keluar. Lalu untuk menu pada admin dapat mengakses bermacam - macam pilihan yang dapat dipilih memungkinkan pengguna admin 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.

### 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 :

### 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.

### 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.

### 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.

### 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.

### Source Code

1. **Login**

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

### Source Code :

| const int max\_kesempatan = 3; *// Maksimal percobaan login*  int kesempatan = 0; *// Menghitung jumlah percobaan login*   while (true) {  if (loggedInUserIndex == -1) {  *// Menu sebelum login*  cout << "\n====================== Selamat Datang ======================" << endl;  cout << "1. Register" << endl;  cout << "2. Login" << endl;  cout << "3. Keluar" << endl;  cout << "===========================================================" << endl;  cout << "Pilih menu (1-3): ";  cin >> pilihan;  cin.ignore();   if (pilihan == 1) {  *// Register User*  if (jumlah\_user < 100) {  cout << "\n====================== Register User ======================" << endl;  cout << "Masukkan Username: ";  getline(cin, users[jumlah\_user].username);  cout << "Masukkan Password: ";  getline(cin, users[jumlah\_user].password);  users[jumlah\_user].isAdmin = false;  jumlah\_user++;  cout << "Registrasi berhasil!\n";  } else {  cout << "Kapasitas user penuh!\n";  }  } else if (pilihan == 2) {  *// Login User*  if (kesempatan < max\_kesempatan) {  cout << "\n========================== Login ==========================" << endl;  string input\_username, input\_password;  cout << "Username: ";  getline(cin, input\_username);  cout << "Password: ";  getline(cin, input\_password);   bool login\_berhasil = false;  for (int i = 0; i < jumlah\_user; i++) {  if (input\_username == users[i].username && input\_password == users[i].password) {  cout << "Login berhasil!\n";  loggedInUserIndex = i;  login\_berhasil = true;  break;  }  }  if (!login\_berhasil) {  kesempatan++;  cout << "Username atau password salah!\n";  cout << "Sisa percobaan login: " << max\_kesempatan - kesempatan << "\n" << endl;  }  } else {  cout << "Anda telah mencoba login sebanyak " << max\_kesempatan << " kali. Program berhenti.\n";  return 0; *// Keluar dari program*  }  } else if (pilihan == 3) {  *// Keluar*  cout << "Terima kasih telah menggunakan program ini!" << endl;  break;  } else {  cout << "Pilihan tidak valid! Silakan coba lagi.\n";  } |
| --- |

1. **Menu**

Terdapat menu pada user yang dapat memilih fitur pada aplikasi

**Source Code**

| ***// Menu User Biasa*  cout << "\n====================== Menu User Biasa ======================" << endl;  cout << "1. Tampilkan Data" << endl;  cout << "2. Pilih Pemain" << endl;  cout << "3. Keluar" << endl;  cout << "=============================================================" << endl;  cout << "Pilih menu (1-3): ";  cin >> pilihan;  cin.ignore();** |
| --- |

Pada menu Admin terdapat beberapa fitur pada aplikasi yang dapat di akses

### Source Code



1. **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 ( Admin ):

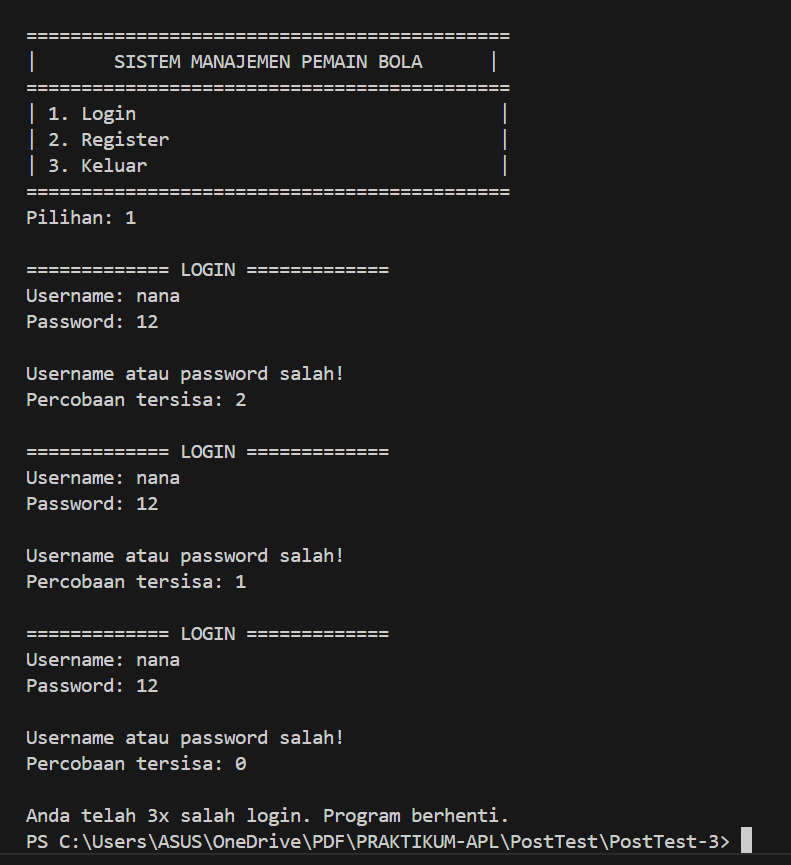
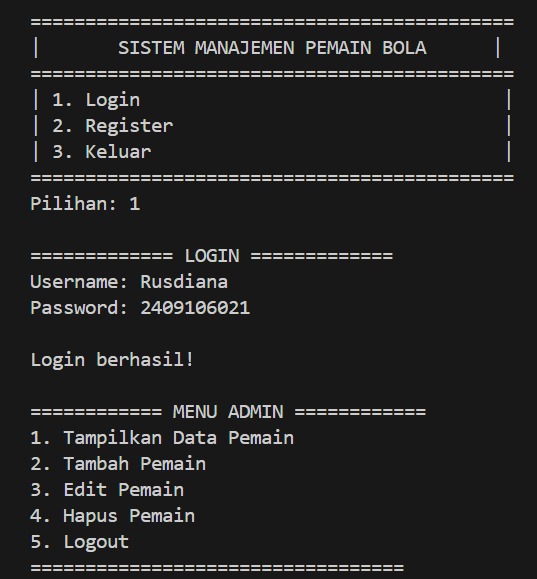
| if (pilihan == 1) {  *// Tampilkan Data*  if (users[loggedInUserIndex].jumlah\_data == 0) {  cout << "Tidak ada data pemain.\n";  } else {  cout << "\n====================== Data Pemain Bola ======================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(20) << "Nomor Punggung" << endl;  cout << "------------------------------------------------------------------" << endl;  for (int i = 0; i < users[loggedInUserIndex].jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << users[loggedInUserIndex].dataPemain[i].nama  << setw(20) << users[loggedInUserIndex].dataPemain[i].posisi  << setw(15) << users[loggedInUserIndex].dataPemain[i].nomorPunggung << endl;  }  cout << endl;  }  } else if (pilihan == 2) {  *// Tambah Data*  if (users[loggedInUserIndex].jumlah\_data < 100) {  cout << "\n====================== Tambah Data Pemain ======================" << endl;  cout << "Masukkan Nama Pemain: ";  getline(cin, users[loggedInUserIndex].dataPemain[users[loggedInUserIndex].jumlah\_data].nama);  cout << "Masukkan Posisi Pemain: ";  getline(cin, users[loggedInUserIndex].dataPemain[users[loggedInUserIndex].jumlah\_data].posisi);  cout << "Masukkan Nomor Punggung: ";  cin >> users[loggedInUserIndex].dataPemain[users[loggedInUserIndex].jumlah\_data].nomorPunggung;  cin.ignore();  users[loggedInUserIndex].jumlah\_data++;  cout << "Data berhasil ditambahkan!\n";  } else {  cout << "Kapasitas data penuh!\n";  }  } else if (pilihan == 3) {  *// Update Data*  if (users[loggedInUserIndex].jumlah\_data > 0) {  cout << "\n====================== Update Data Pemain ======================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(15) << "Nomor Punggung" << endl;  cout << "------------------------------------------------------------------" << endl;  for (int i = 0; i < users[loggedInUserIndex].jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << users[loggedInUserIndex].dataPemain[i].nama  << setw(20) << users[loggedInUserIndex].dataPemain[i].posisi  << setw(15) << users[loggedInUserIndex].dataPemain[i].nomorPunggung << endl;  }  cout << endl;   int index;  cout << "Masukkan nomor pemain yang ingin diupdate: ";  cin >> index;  cin.ignore();  if (index >= 1 && index <= users[loggedInUserIndex].jumlah\_data) {  cout << "Masukkan Nama Pemain Baru: ";  getline(cin, users[loggedInUserIndex].dataPemain[index - 1].nama);  cout << "Masukkan Posisi Pemain Baru: ";  getline(cin, users[loggedInUserIndex].dataPemain[index - 1].posisi);  cout << "Masukkan Nomor Punggung Baru: ";  cin >> users[loggedInUserIndex].dataPemain[index - 1].nomorPunggung;  cin.ignore();  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 (users[loggedInUserIndex].jumlah\_data > 0) {  cout << "\n====================== Hapus Data Pemain ======================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(15) << "Nomor Punggung" << endl;  cout << "------------------------------------------------------------------" << endl;  for (int i = 0; i < users[loggedInUserIndex].jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << users[loggedInUserIndex].dataPemain[i].nama  << setw(20) << users[loggedInUserIndex].dataPemain[i].posisi  << setw(15) << users[loggedInUserIndex].dataPemain[i].nomorPunggung << endl;  }  cout << endl;   int index;  cout << "Masukkan nomor pemain yang ingin dihapus: ";  cin >> index;  cin.ignore();  if (index >= 1 && index <= users[loggedInUserIndex].jumlah\_data) {  for (int i = index - 1; i < users[loggedInUserIndex].jumlah\_data - 1; i++) {  users[loggedInUserIndex].dataPemain[i] = users[loggedInUserIndex].dataPemain[i + 1];  }  users[loggedInUserIndex].jumlah\_data--;  cout << "Data berhasil dihapus!\n";  } else {  cout << "Nomor pemain tidak valid!\n";  }  } else {  cout << "Tidak ada data untuk dihapus!\n";  }  } else if (pilihan == 5) {  *// Keluar*  loggedInUserIndex = -1;  cout << "Logout berhasil!\n";  } else {  cout << "Pilihan tidak valid! Silakan coba lagi.\n";  } |
| --- |

### Source Code ( User Baru )

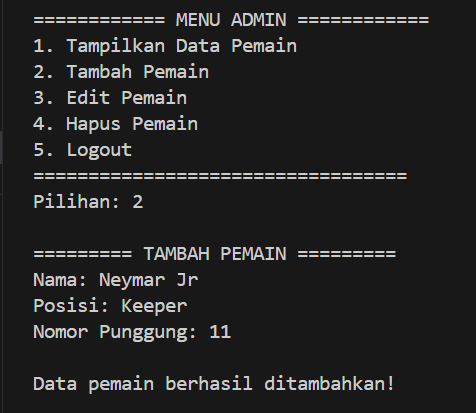
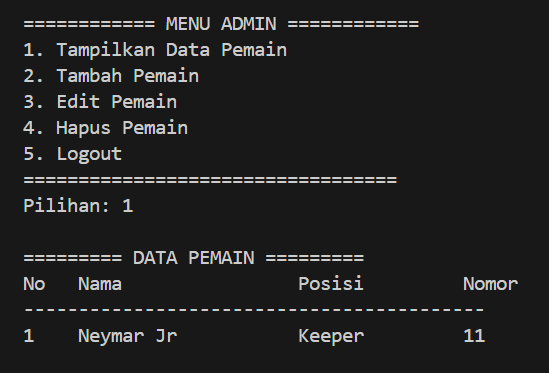
| *// Menu User Biasa*  cout << "\n====================== Menu User Biasa ======================" << endl;  cout << "1. Tampilkan Data" << endl;  cout << "2. Pilih Pemain" << endl;  cout << "3. Keluar" << endl;  cout << "=============================================================" << endl;  cout << "Pilih menu (1-3): ";  cin >> pilihan;  cin.ignore();   if (pilihan == 1) {  *// Tampilkan Data*  if (users[loggedInUserIndex].jumlah\_data == 0) {  cout << "Tidak ada data pemain.\n";  } else {  cout << "\n====================== Data Pemain Bola ======================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(20) << "Nomor Punggung" << endl;  cout << "------------------------------------------------------------------" << endl;  for (int i = 0; i < users[loggedInUserIndex].jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << users[loggedInUserIndex].dataPemain[i].nama  << setw(20) << users[loggedInUserIndex].dataPemain[i].posisi  << setw(15) << users[loggedInUserIndex].dataPemain[i].nomorPunggung << endl;  }  cout << endl;  }  } else if (pilihan == 2) {  *// Pilih Pemain*  if (users[loggedInUserIndex].jumlah\_data > 0) {  cout << "\n====================== Pilih Pemain ======================" << endl;  cout << setw(5) << "No" << setw(20) << "Nama Pemain" << setw(20) << "Posisi" << setw(15) << "Nomor Punggung" << endl;  cout << "------------------------------------------------------------------" << endl;  for (int i = 0; i < users[loggedInUserIndex].jumlah\_data; i++) {  cout << setw(5) << i + 1 << setw(20) << users[loggedInUserIndex].dataPemain[i].nama  << setw(20) << users[loggedInUserIndex].dataPemain[i].posisi  << setw(15) << users[loggedInUserIndex].dataPemain[i].nomorPunggung << endl;  }  cout << endl;   int index;  cout << "Masukkan nomor pemain yang ingin dipilih: ";  cin >> index;  cin.ignore();  if (index >= 1 && index <= users[loggedInUserIndex].jumlah\_data) {  cout << "Anda memilih pemain: " << users[loggedInUserIndex].dataPemain[index - 1].nama << endl;  } else {  cout << "Nomor pemain tidak valid!\n";  }  } else {  cout << "Tidak ada data pemain untuk dipilih!\n";  }  } else if (pilihan == 3) {  *// Keluar*  loggedInUserIndex = -1;  cout << "Logout berhasil!\n";  } else {  cout << "Pilihan tidak valid! Silakan coba lagi.\n";  }  }  }  }   return 0; } |
| --- |

### 4. Hasil Output

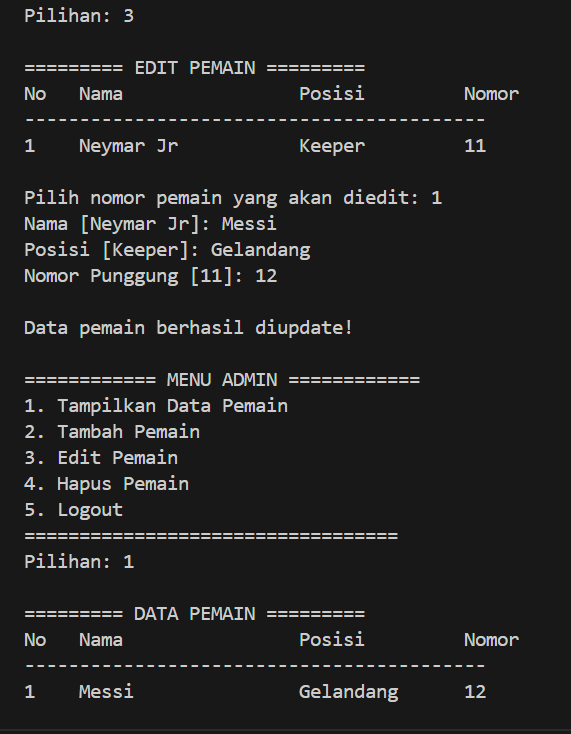
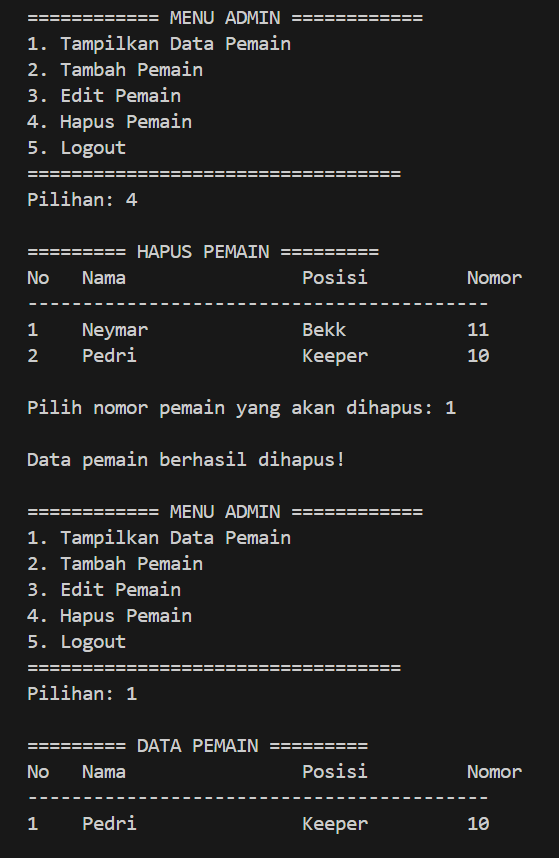
1. Menu Admin

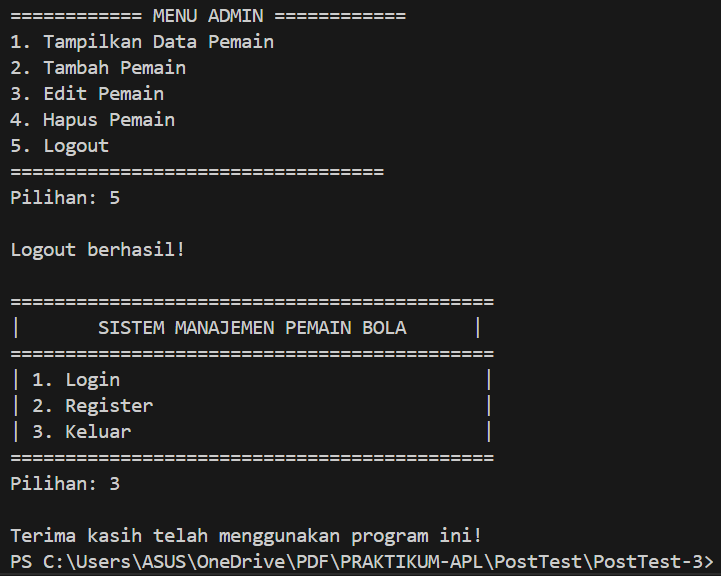
Gambar 4.1.1 Kondisi Gagal Login Gambar 4..1.2 Tampilan Login Admin

Gambar 4.1.3 Menu Tambah Pemain Gambar 4.1.4 Menu Tampilan Data

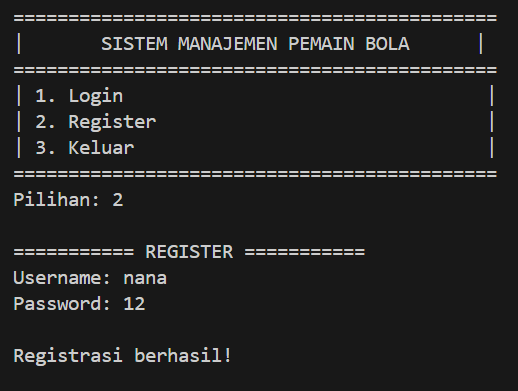
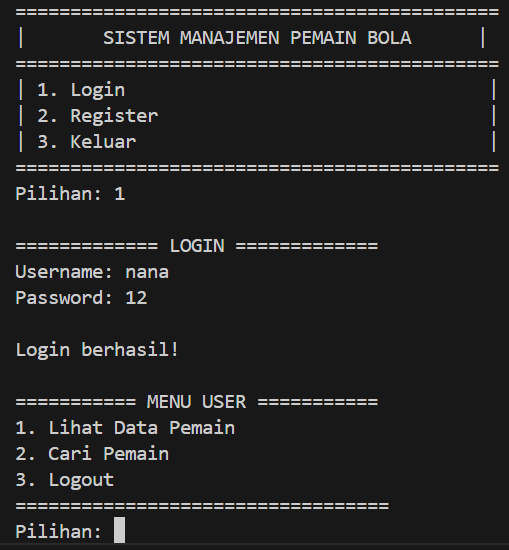
 

Gambar 4.1.5 Menu Edit Pemain Gambar 4.1.6 Menu Hapus Pemain

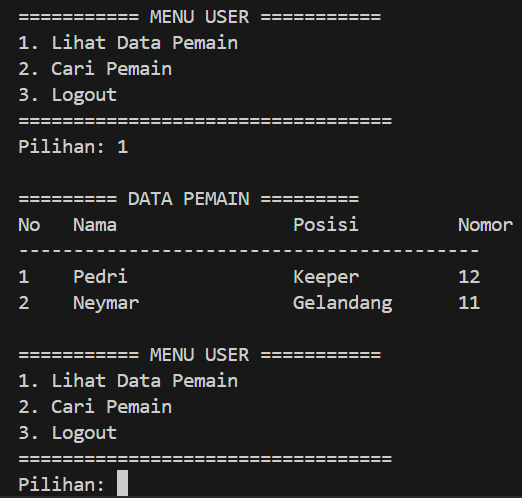
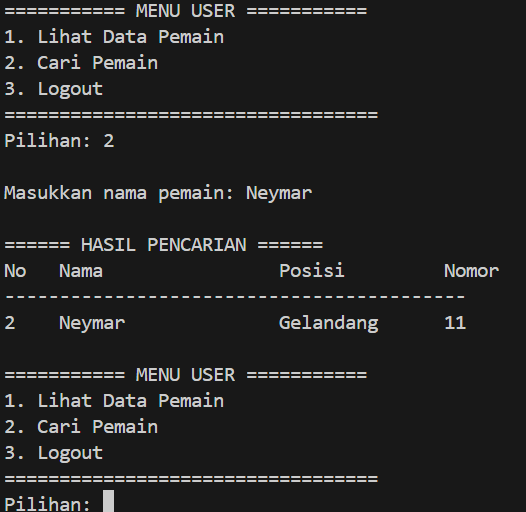


Gambar 4.1.7 Menu Logout

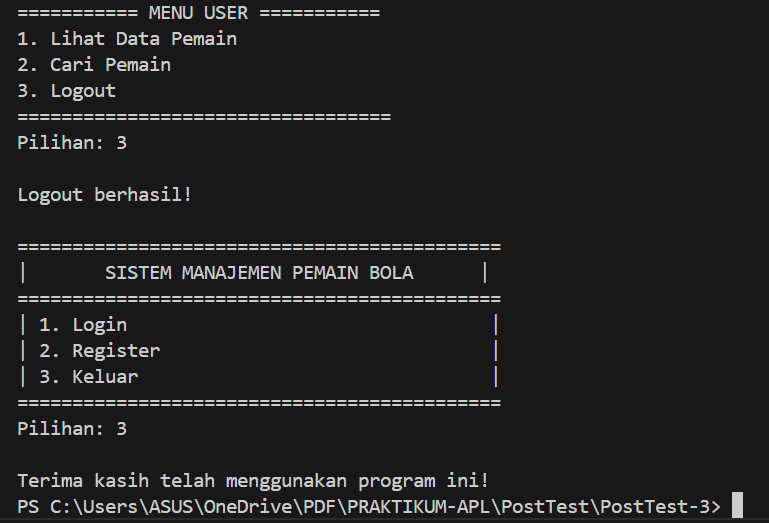
1. Menu User Baru

Gambar 4.2.1 Registrasi Username Baru Gambar 4.2.2 Login Sebagai User Baru

Gambar 4.2.3 Tampilan Lihat Data Pemain Gambar 4.2.3 Menu Cari Pemain



Gambar 4.2.4 Menu Logout

1. **Langkah-Langkah GIT**

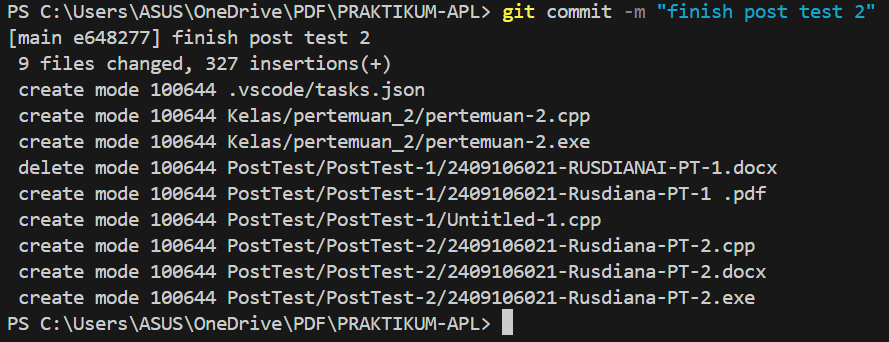
# Git add

## 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

Gambar 5. 1 Git\_add

1. **Git Commit**

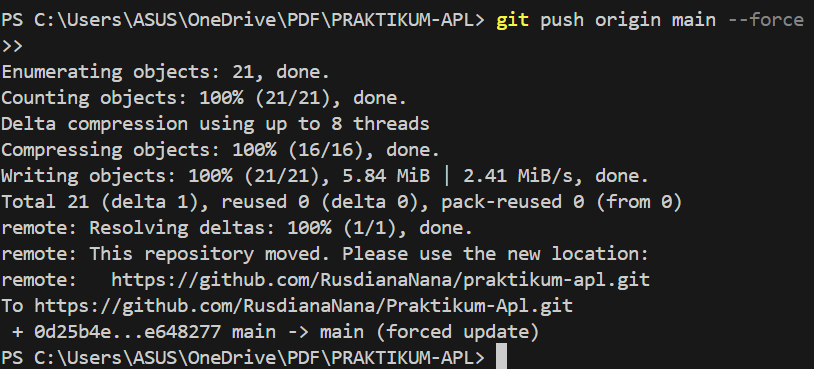
## Git commit adalah perintah Git yang digunakan untuk menyimpan perubahan yang telah ditambahkan ke staging area ke dalam repository. Setiap commit akan memiliki hash unik, pesan commit, dan menyimpan snapshot dari perubahan yang dilakukan.



Gambar 5. 2 Git\_Commit

1. **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.



Gambar 5. 3 Git\_Push