

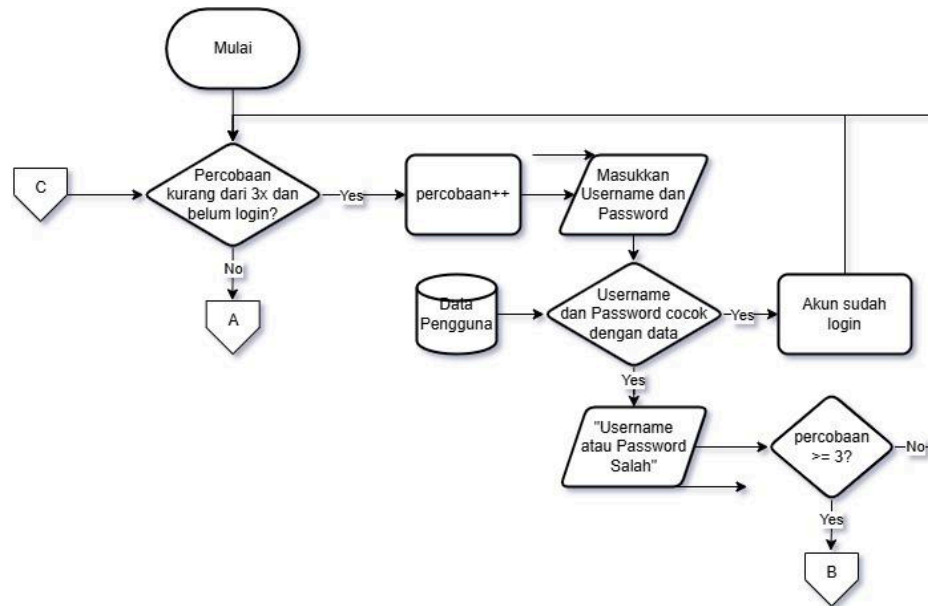
**LAPORAN PRAKTIKUM**  
**POSTTEST 6**  
**ALGORITMA PEMROGRAMAN LANJUT**



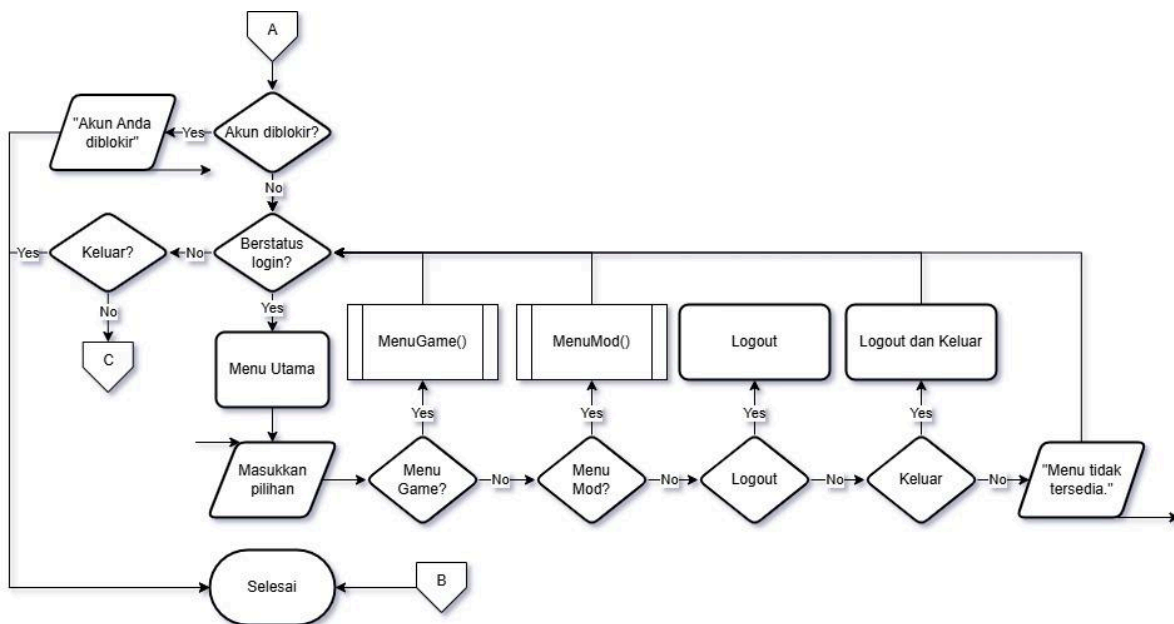
**Disusun oleh:**  
**Much. Trigusni Hermawan (2409106060)**  
**Kelas (B1 '24)**

**PROGRAM STUDI INFORMATIKA**  
**UNIVERSITAS MULAWARMAN**  
**SAMARINDA**  
**2025**

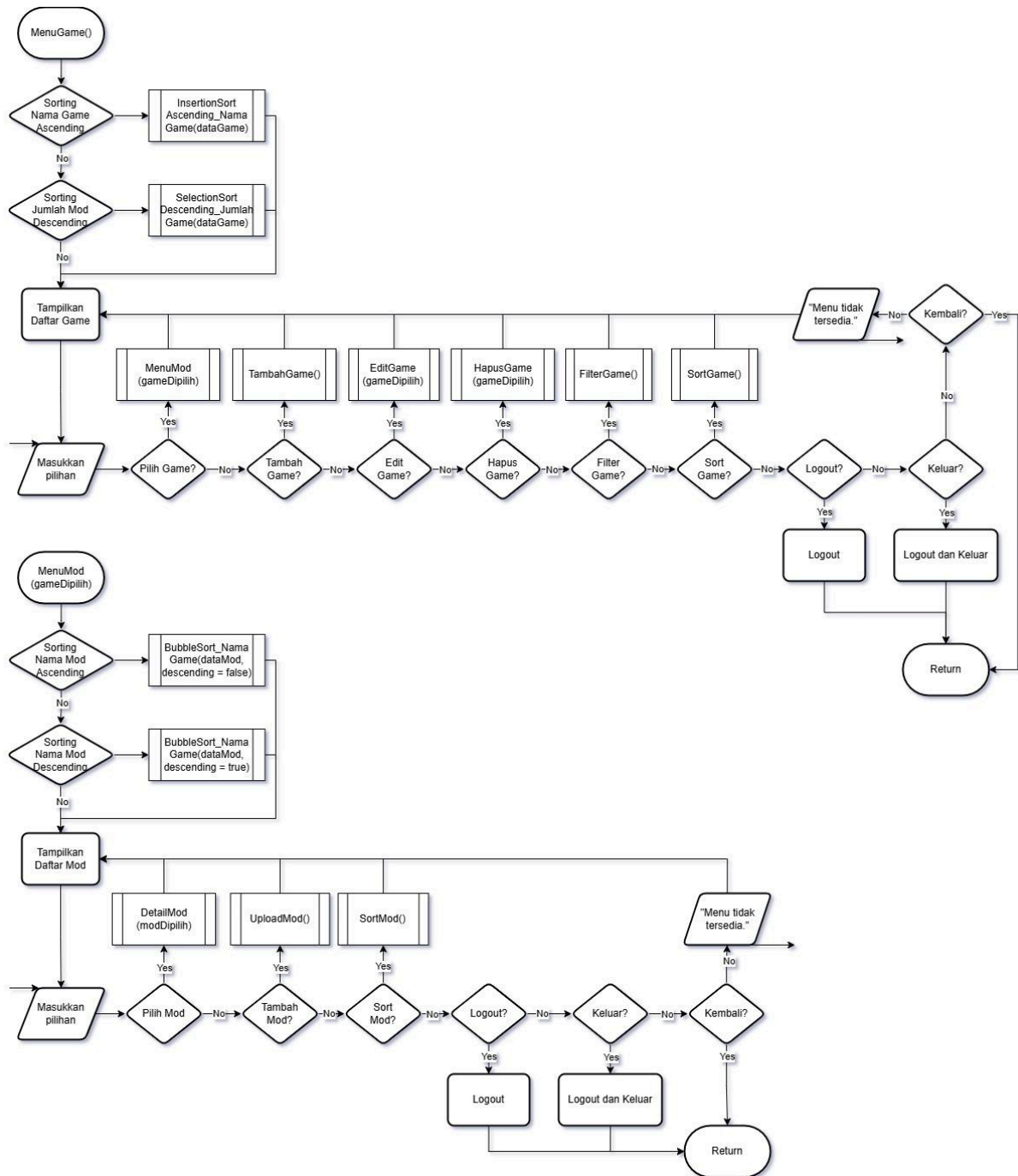
## 1. Flowchart



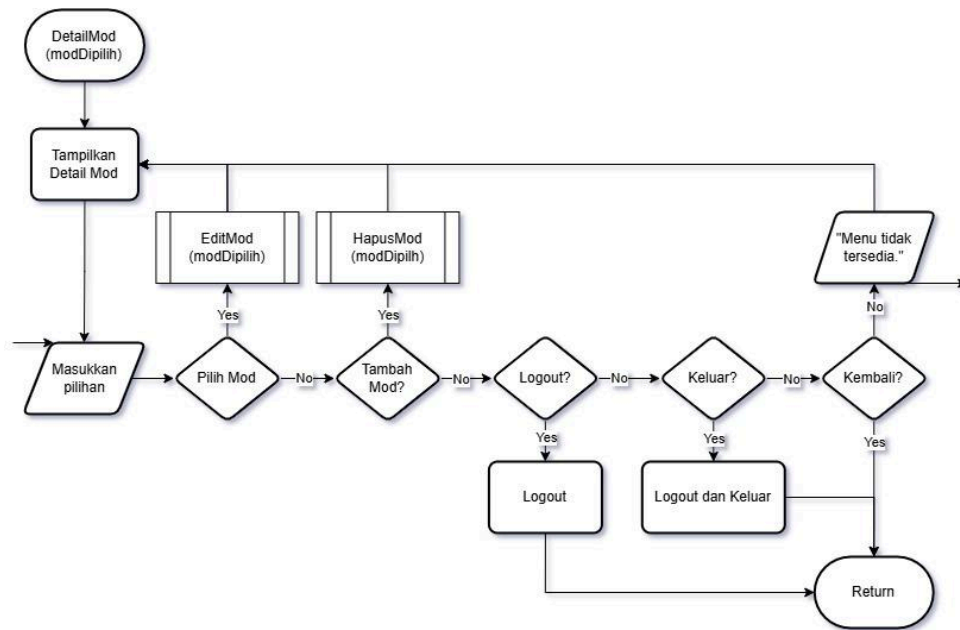
Gambar 1.1 Main Flow Bagian 1



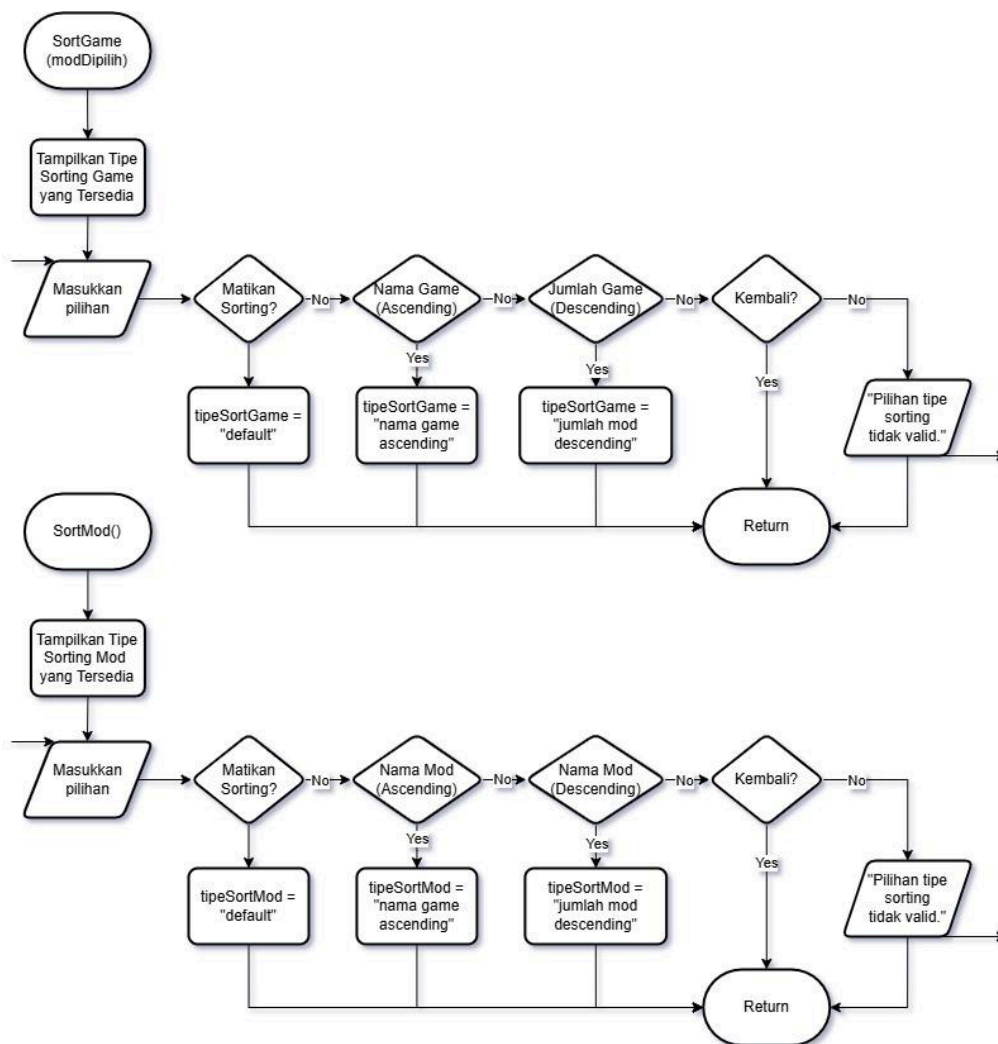
Gambar 1.2 Main Flow Bagian 2



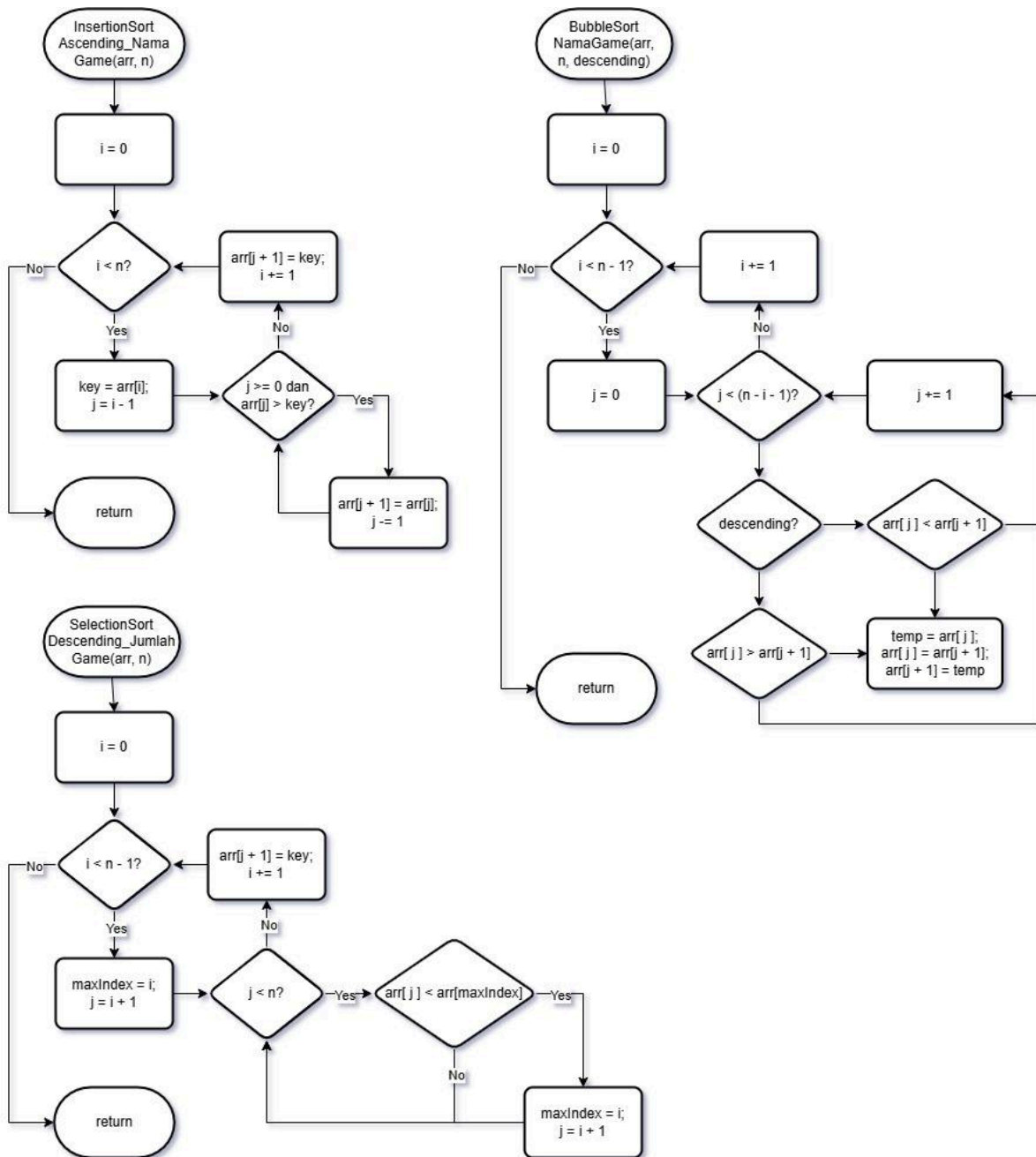
Gambar 1.3 Predefined Process *MenuGame()* dan *MenuMod()*



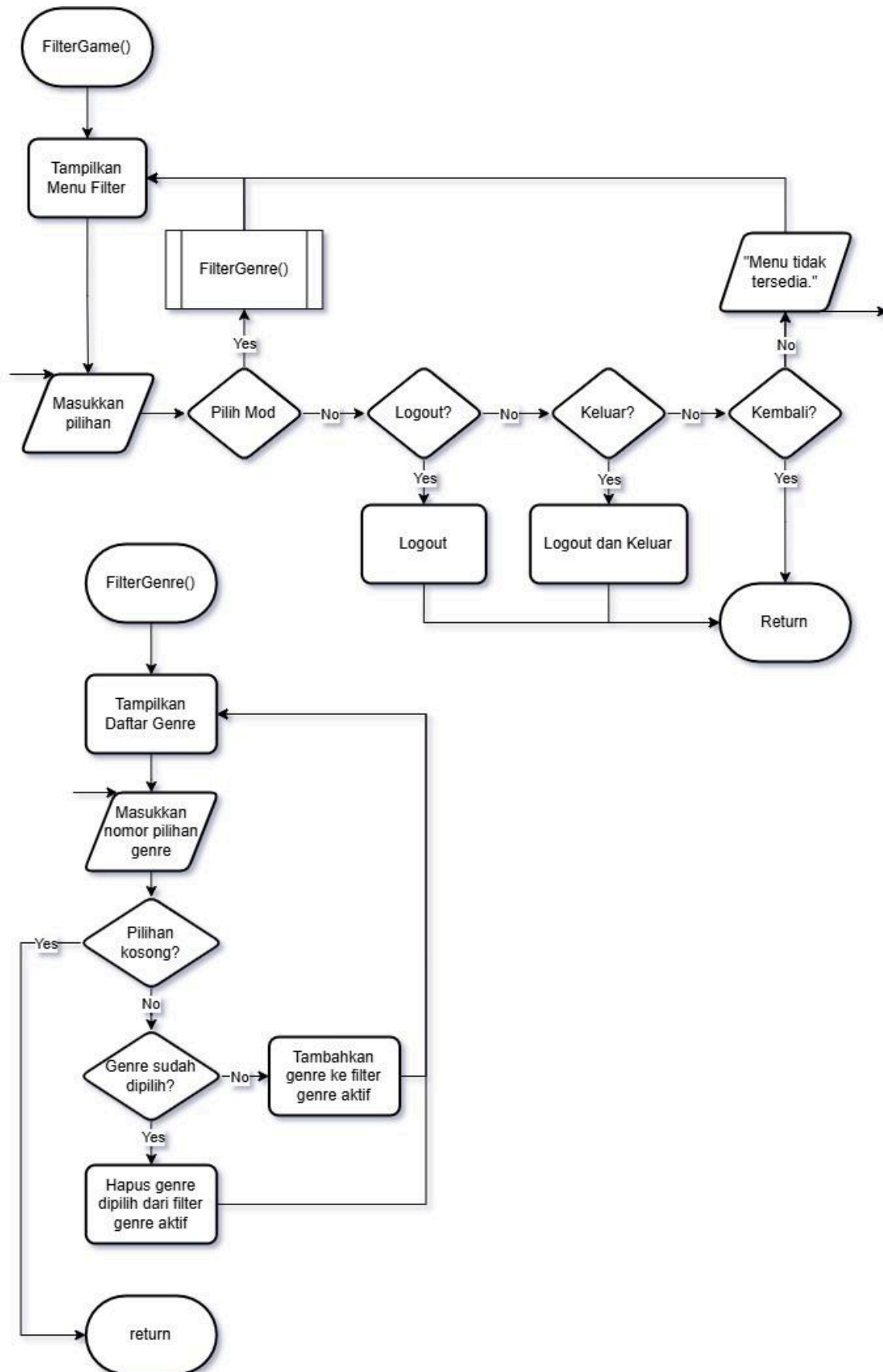
Gambar 1.4 Predefined Process *DetailMod()*



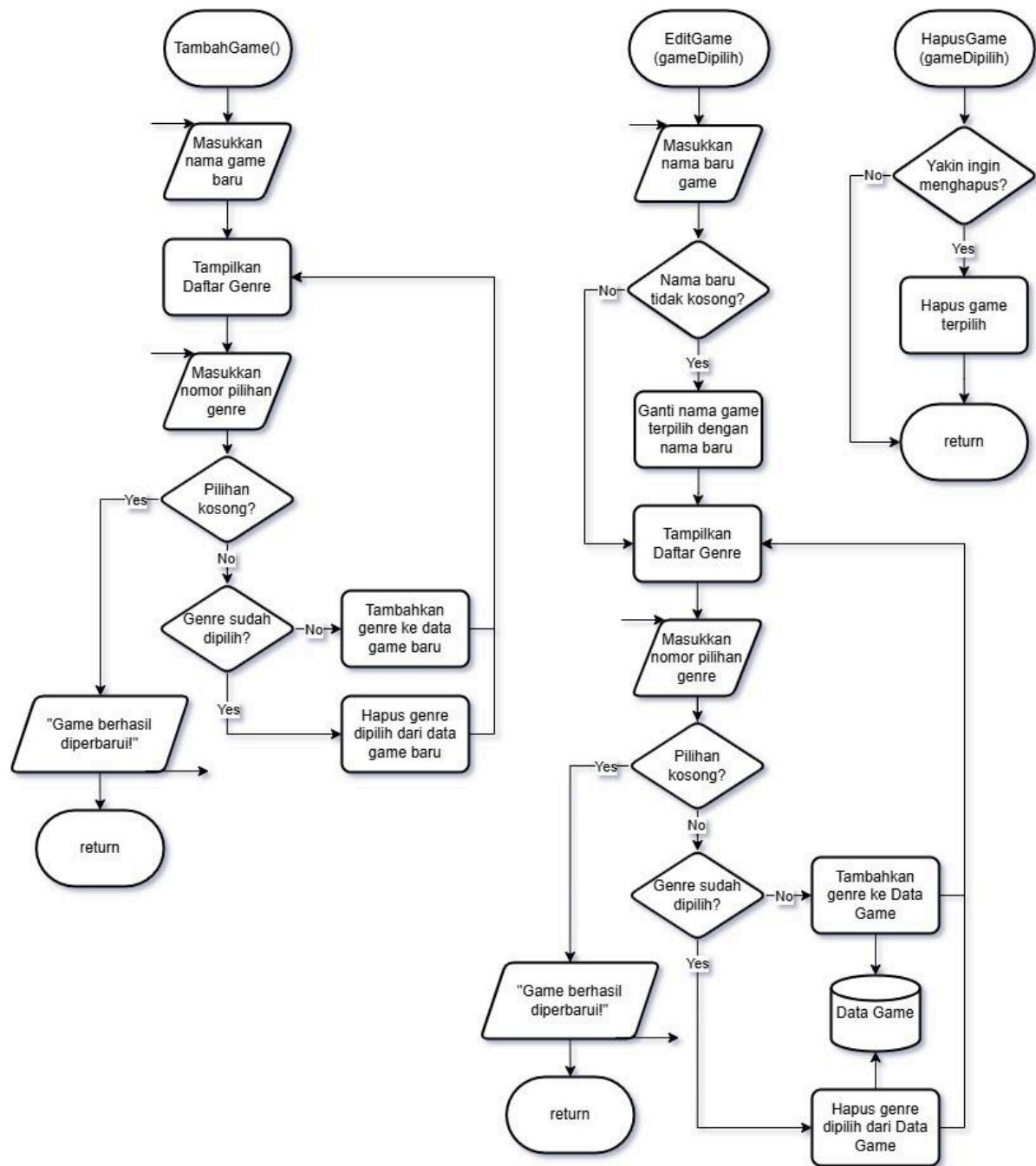
Gambar 1.5 Predefined Process *SortGame()*



Gambar 1.6 Predefined Process *InsertionSortAscendingNamaGame()*, *SelectionSortDescending\_JumlahGame()*, dan *BubbleSort\_NamaMod()*

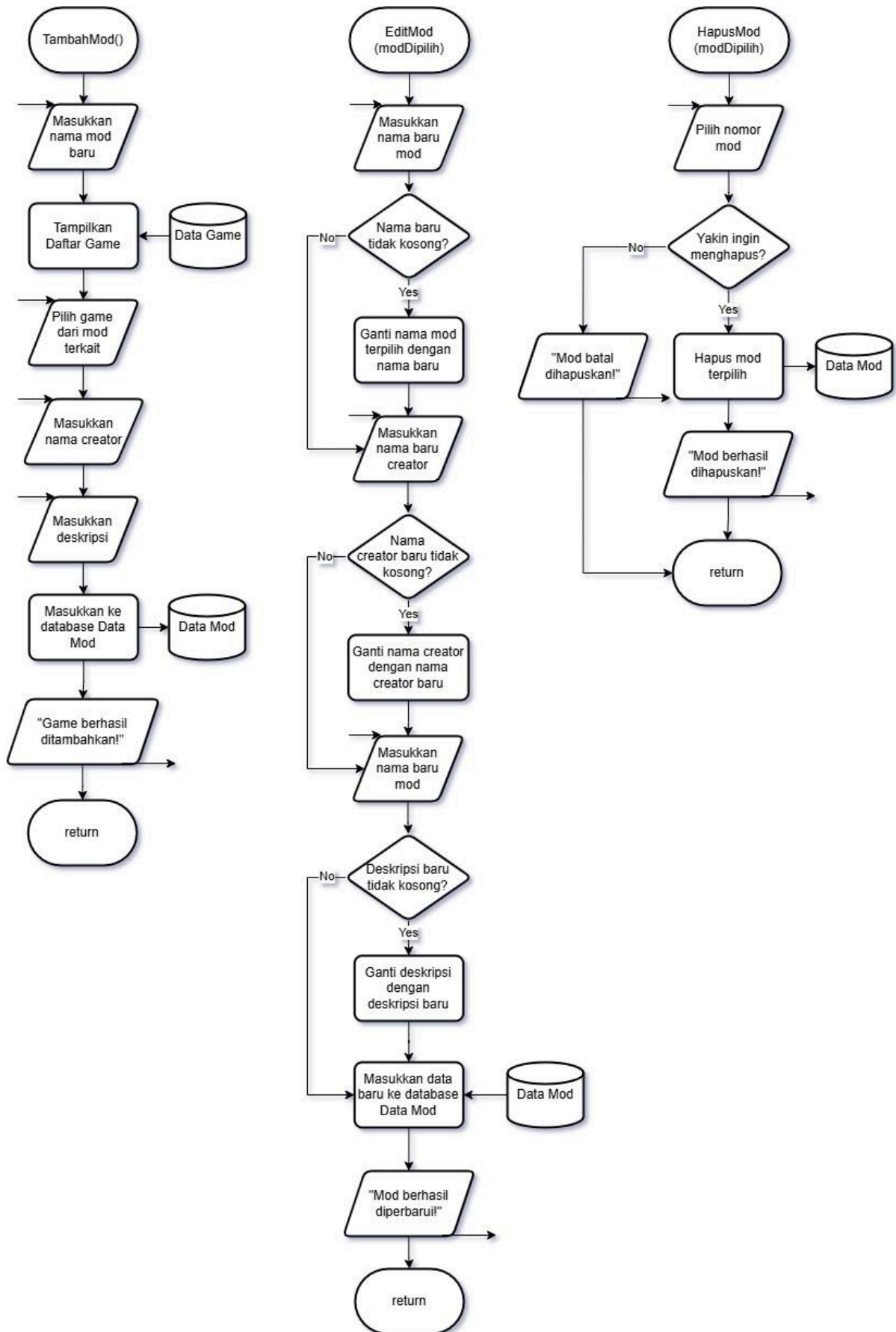


Gambar 1.7 Predefined Process Menu *FilterGame()* dan *FilterGenre()*



Gambar 1.8 Predefined Process *TambahGame()*, *EditGame()*, dan *HapusGame()* (CRUD Game)





Gambar 1.9 Predefined Process *TambahMod()*, *EditMod()*, dan *HapusMod()* (CRUD Mod)



## 2. Analisis Program

### 2.1 Deskripsi Singkat Program

Program Pengelolaan *Video Game Modding* ini dibuat bertujuan sebagai sebuah wadah bagi para komunitas penggemar sebuah *video game* membagikan dan mencari konten lebih dari para penggemar lainnya. Program ini diharapkan dapat memudahkan para *modder* menjadi lebih terbantu untuk membagikan kreasi *mod* buatannya dan begitu juga para *player* yang ingin dengan mudah mencari sekumpulan *mod* untuk dimainkan.

## 3. Source Code

### A. PilihanCheckbox

Salah satu fungsi penting yang akan digunakan di beberapa fungsi lainnya yaitu pilihan checkbox, seperti untuk memilih beberapa filter genre.

Source Code:

```
void PilihanCheckbox(string selectedData[], int &selectedDataCount, string
options[], int optionCount, string title)
{
    string _temp;
    int selectedIndex = -1;

    while (true)
    {
        cout << title << endl;
        for (int i = 0; i < optionCount; i++)
        {
            bool isSelected = false;

            for (int j = 0; j < selectedDataCount; j++)
            {
                if (options[i] == selectedData[j])
                {
                    isSelected = true;
                    break;
                }
            }
            cout << i + 1 << ". " << options[i] << (isSelected ? " (Dipilih)" :
            "") << endl;
```

```

    }

    cout << endl;
    cout << "(Kosongkan input untuk mengakhiri pilihan.)" << endl;
    cout << "Pilih pilihan: ";
    getline(cin, _temp);

    if (_temp == "")
        break;

    for (int i = 0; i < optionCount; i++)
    {
        if (to_string(i + 1) == _temp)
        {
            selectedIndex = i;
            break;
        }
    }

    if (selectedIndex >= 0 && selectedIndex < optionCount)
    {
        bool alreadySelected = false;

        for (int i = 0; i < selectedDataCount; i++)
        {
            if (selectedData[i] == options[selectedIndex])
            {
                for (int j = i; j < selectedDataCount - 1; j++)
                {
                    selectedData[j] = selectedData[j + 1];
                }

                selectedDataCount--;
                alreadySelected = true;
                break;
            }
        }

        if (!alreadySelected && selectedDataCount < SMOL_DATA)
        {
            selectedData[selectedDataCount] = options[selectedIndex];
            selectedDataCount++;
        }
    }
    else
    {
        cout << "Pilihan tidak valid!" << endl;
    }
}
}

```

## B. Insertion Sort Ascending untuk Nama Game

Insertion Sort *ascending* untuk mengurutkan daftar game berdasarkan namanya (A-Z).

### Source Code:

```
void InsertionSortAscending_NamaGame(Game *arr, int panjang)
{
    for (int i = 1; i < panjang; i++)
    {
        Game key = arr[i];
        int j = i - 1;

        while (j >= 0 && arr[j].nama.compare(key.nama) > 0)
        {
            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
    }
}
```

## C. Selection Sort Ascending untuk Jumlah Mod Game

Selection Sort *descending* untuk mengurutkan daftar game berdasarkan jumlah modnya (... 7, 4, 2, ...).

### Source Code:

```
void SelectionSortDescending_JumlahModGame(Game *arr, int panjang)
{
    for (int i = 0; i < panjang - 1; i++)
    {
        int maxIndex = i;
        for (int j = i + 1; j < panjang; j++)
        {
            if (arr[j].jumlahMod < arr[maxIndex].jumlahMod)
            {
                maxIndex = j;
            }
        }

        Game temp = arr[i];
        arr[i] = arr[maxIndex];
        arr[maxIndex] = temp;
    }
}
```

#### D. Bubble Sort Ascending untuk Nama Mod

Bubble Sort *ascending/descending* untuk mengurutkan daftar mod berdasarkan nama mod itu sendiri.

##### Source Code:

```
void BubbleSort_NamaMod(Mod *arr, int panjang, bool descending)
{
    for (int i = 0; i < panjang; i++)
    {
        for (int j = 0; j < (panjang - i - 1); j++)
        {
            int strCompare = arr[j].nama.compare(arr[j + 1].nama);
            if (!descending ? strCompare > 0 : strCompare < 0)
            {
                // Menukar elemen jika elemen sebelumnya Lebih besar
                Mod temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}
```

#### E. Login

Login digunakan untuk pengguna melanjutkan ke Menu Utama program. Di proses login, pengguna perlu memasukkan username dan password dari akun yang pernah dibuat pada saat registrasi. Maksimal percobaan hingga program pengguna dihentikan adalah sebanyak 3 kali.

##### Source Code:

```
void Login(int &batasPercobaan, int &percobaanGagal, User *users, int userCount,
User *currentUser)
{
    string inputUsername = "";
    string inputPassword = "";

    do
    {
        cout << "Username\t: ";
        getline(cin, inputUsername);
        if (inputUsername == "")
            break;
    }
```

```

    cout << "Password\t: ";
    getline(cin, inputPassword);
    if (inputPassword == "")
        break;

    // Proses validasi login
    *currentUser = GetUserLogin(users, userCount, inputUsername,
inputPassword);

    if (currentUser->id > 0)
    {
        cout << "Login berhasil!" << endl;
    }
    else
    {
        percobaanGagal++;
        cout << "Username atau Password salah!" << endl;
    }
} while (percobaanGagal < batasPercobaan && currentUser->id <= 0);
}

```

## F. Registrasi

Registrasi digunakan pengguna untuk membuat akun baru. Dalam proses registrasi pengguna perlu memasukkan username dan password yang akan digunakan dalam proses login.

### Source Code:

```

void Register(User *userPtr, int &userCount)
{
    User newUser;
    string _temp = "";

    if (userCount < MAX_DATA)
    {
        // Proses registrasi dapat berjalan jika user belum mencapai maks
        cout << "Registrasi" << endl;
        cout << endl;
        cout << "Username\t: ";
        getline(cin, newUser.username);

        for (int i = 0; i < userCount; i++)
        {
            if (newUser.username != "" && newUser.username ==
userPtr[i].username)
            {
                newUser.username = "";
                cout << "Username sudah ada!";
            }
        }
    }
}

```

```

        getline(cin, _temp);
        return;
    }
}

if (newUser.username == "")
    return;

cout << "Password\t: ";
getline(cin, newUser.password);

if (newUser.password == "")
    return;

// Default Value
newUser.id = userCount + 1;
newUser.level = "user";

// Menambahkan user baru
userPtr[userCount] = newUser;
userCount++;

cout << "Registrasi berhasil!";
getline(cin, _temp);
cout << endl;
}
else
{
    cout << "Registrasi gagal! Coba dalam beberapa saat lagi.";
    getline(cin, _temp);
}
}
}

```

## G. Sistem Login

Pada awal program, pengguna akan berada dalam menu Sistem Login, di mana pengguna dapat memilih untuk melakukan login atau registrasi jika belum memiliki akun.

### Source Code:

```

void MenuSistemLogin(int &batasPercobaan, int &percobaanGagal, User *usersPtr,
int &userCount, User *currentUser)
{
    string _temp;

    // Sistem Login
    Border("Sistem Pengelolaan Mod");

```

```

cout << "1 > Login" << endl;
cout << "2 > Register" << endl;
Border("-", 106);
cout << "Q > Keluar" << endl;
cout << endl;
cout << "Pilih Menu\t: ";
getline(cin, _temp);
Border("-", 106);
if (_temp == "1")
{
    Login(batasPercobaan, percobaanGagal, usersPtr, userCount, currentUser);
}
else if (_temp == "2")
{
    Register(usersPtr, userCount);
}
else if (_temp != "Q" && _temp != "q")
{
    cout << "Menu tidak ditemukan!";
    getline(cin, _temp);
}
}

```

## H. Menu Utama

Menu Utama menampilkan daftar dari fitur utama program yang dapat pengguna pilih, yaitu “Menu Game” dan “Menu Mod.” Menu ini diakses setelah pengguna berhasil melakukan login.

### Source Code:

```

void MenuUtama(char &halaman, string &pilihan_menu, User *currentUser)
{
    string _temp = "";

    // Menu Utama
    cout << "Menu Utama" << endl;
    Border("-", 106);
    cout << "Selamat datang, " << currentUser->username << "!" << endl;
    cout << endl;
    cout << "1 > Games" << endl;
    cout << "2 > Mods" << endl;
    Border("-", 106);
    cout << "Q > Keluar | L > Logout" << endl;
    cout << endl;
    cout << "Pilih Menu\t: ";
    getline(cin, _temp);
}

```



```

    if (_temp == "1")
    {
        halaman = '1';
    }
    else if (_temp == "2")
    {
        halaman = '2';
    }
    else if ((_temp == "Q" || _temp == "q") || (_temp != "B" || _temp != "b") ||
(_temp == "L" || _temp == "l"))
    {
        pilihan_menu = _temp;
    }
    else
    {
        cout << "Menu tidak ditemukan!";
        getline(cin, _temp);
    }
}

```

## I. Tambah Game

Fitur “Tambah Game” memungkinkan admin untuk menambahkan *game* baru ke dalam daftar game. Dalam proses penambahan game, masukan yang perlu dipenuhi terdiri dari nama dan genre game.

### Source Code:

```

void TambahGame(Game *gamesPtr, int &gameCount, GabunganArray *dataArrayPtr, int
&genreCount)
{
    string _temp = "";
    Game gameBaru;

    if (gameCount <= MAX_DATA)
    {
        Border("-", 106);
        cout << "Tambah Game" << endl;
        Border("-", 106);

        // Input Nama Game
        cout << "Nama Game\t: ";
        getline(cin, gameBaru.nama);

        for (int i = 0; i < gameCount; i++)
        {
            if (gameBaru.nama != "" && gamesPtr[i].nama == gameBaru.nama)
            {

```

```

        gameBaru.nama = "";
        cout << "Game sudah ada!";
        getline(cin, _temp);
        break;
    }
}

if (gameBaru.nama == "")
    return;

// Input Genre
string genres[SMOL_DATA];
for (int i = 0; i < genreCount; i++)
{
    genres[i] = dataArrayPtr[i].genre;
}
PilihanCheckbox(gameBaru.genre, gameBaru.genreCount, genres, genreCount,
"Genre");
cout << endl;

// Prsoses penambahan game
gamesPtr[gameCount] = gameBaru;
gameCount++;
cout << "Game berhasil ditambahkan!";
getline(cin, _temp);
}
else
{
    cout << "Game penuh! Tidak dapat menambah game lagi.";
    getline(cin, _temp);
}
}

```

## J. Edit Game

Fitur “Edit Game” memungkinkan admin untuk memperbarui data game yang sudah terdaftar, seperti mengganti nama atau genre dari suatu game.

### Source Code:

```

void EditGame(Game gameListMenu[], int gameListMenuCount, Game *gamesPtr, int
&gameCount, GabunganArray *dataArrayPtr, int &genreCount)
{
    string _temp = "";
    int _selectedIndex = -1;

    // Input pilihan game
    cout << "Pilih Game\t: ";

```

```

getline(cin, _temp);

if (_temp == "")
    return;

for (int i = 0; i < gameListMenuCount; i++)
{
    if (to_string(i + 1) == _temp)
    {
        for (int j = 0; j < gameCount; j++)
        {
            if (gameListMenu[i].nama == gamesPtr[j].nama)
            {
                _selectedIndex = j;
                break;
            }
        }
        break;
    }
}

if (gameCount > _selectedIndex >= 0)
{
    // Lanjutkan proses edit game jika indeks valid
    Game gameBaru = gamesPtr[_selectedIndex];
    Border("-", 106);
    cout << "Edit Game \"" << gamesPtr[_selectedIndex].nama << "\"" << endl;
    Border("-", 106);

    // Input Nama Game
    cout << "Nama Game Baru\t: ";
    getline(cin, _temp);
    cout << endl;

    // Validasi nama game
    for (int i = 0; i < gameCount; i++)
    {
        if (_temp != "" && gamesPtr[i].nama == _temp && gameBaru.nama !=
_temp)
        {
            _temp = "";
            cout << "Game sudah ada!";
            getline(cin, _temp);
            break;
        }
    }

    if (_temp != "")
        gameBaru.nama = _temp;

    // Input Genre

```

```

        string genres[SMOL_DATA];
        for (int i = 0; i < genreCount; i++)
        {
            genres[i] = dataArrayPtr[i].genre;
        }
        PilihanCheckbox(gameBaru.genre, gameBaru.genreCount, genres, genreCount,
"Genre");
        cout << endl;

        // Proses perubahan data game
        gamesPtr[_selectedIndex] = gameBaru;
        cout << "Game berhasil diperbarui!";
        getline(cin, _temp);
    }
    else
    {
        cout << "Game tidak ditemukan!";
        getline(cin, _temp);
    }
}

```

## K. Hapus Game

Fitur “Hapus Game” memungkinkan admin untuk menghapus suatu game yang telah terdaftar.

### Source Code:

```

void HapusGame(Game gameListMenu[], int gameListMenuCount, Game *gamesPtr, int
&gameCount, Mod *mods, int &modCount)
{
    string _temp = "";
    int _selectedIndex = -1;

    // Input pilihan game
    cout << "Pilih Game\t: ";
    getline(cin, _temp);

    if (_temp == "")
        return;

    for (int i = 0; i < gameListMenuCount; i++)
    {
        if (to_string(i + 1) == _temp)
        {
            for (int j = 0; j < gameCount; j++)
            {
                if (gameListMenu[i].nama == gamesPtr[j].nama)

```

```

        {
            _selectedIndex = j;
            break;
        }
    }
    break;
}

if (gameCount > _selectedIndex >= 0)
{
    // Lanjutkan proses penghapusan game jika indeks valid
    cout << endl;
    cout << "Apakah Anda yakin ingin menghapus \"" <<
gamesPtr[_selectedIndex].nama << "\"? [Y/N]\t: ";
    getline(cin, _temp);

    if (_temp == "Y" || _temp == "y")
    {
        // Proses penghapusan game
        for (int i = _selectedIndex; i < gameCount - 1; i++)
        {
            gamesPtr[i] = gamesPtr[i + 1];
        }
        gameCount--;
        cout << "Game berhasil dihapus!";
        getline(cin, _temp);

        for (int i = 0; i < modCount; i++)
        {
            if (mods[i].game->nama == gamesPtr[_selectedIndex].nama)
            {
                mods[i].game = new Game();
            }
        }
    }
    else
    {
        // Batalkan proses penghapusan game
        cout << "Game dibatalkan dihapus!";
        getline(cin, _temp);
    }
}
else
{
    cout << "Game tidak ditemukan!";
    getline(cin, _temp);
}
}
}

```

## L. Sort Game

Fitur “Sort Game”, memungkinkan pengguna dapat memilih tipe sorting data yang ingin ditampilkan dalam daftar game.

### Source Code:

```
void SortGame(int &gameSortType)
{
    string _temp = "";

    cout << endl;
    cout << "0 > Matikan Sorting" << endl;
    cout << "1 > Nama Game (Ascending)" << endl;
    cout << "2 > Jumlah Game (Descending)" << endl;
    cout << endl;
    cout << "Pilih Tipe Sorting:";
    getline(cin, _temp);

    if (_temp == "0")
        gameSortType = 0;
    else if (_temp == "1")
        gameSortType = 1;
    else if (_temp == "2")
        gameSortType = 2;
    else
        cout << "Pilihan tipe sorting valid!" << endl;
}
```

## M. Menu Game

Di “Menu Game”, program menampilkan daftar game yang tersedia. Program juga menyediakan fitur bagi admin untuk menambahkan, memperbarui, dan menghapus *game*.

### Source Code:

```
void MenuGame(char &halaman, string &pilihan_menu, Game &selectedGame, string
selectedGenres[], int &selectedGenreCount, Game *gamesPtr, int &gameCount, Mod
*modsPtr, int &modCount, GabunganArray *dataArrayPtr, int &genreCount, User
*currentUser, int &gameSortType)
{
    string _temp;
    int i;
```

```

Game gameListMenu[MAX_DATA];
int gameListMenuCount = 0;

selectedGame = Game();

// Menduplikat data game
for (int i = 0; i < gameCount; i++)
{
    gameListMenu[gameListMenuCount] = gamesPtr[i];
    gameListMenuCount++;
}

// Proses pemfilteran game
i = 0;
while (i < gameListMenuCount && selectedGenreCount > 0)
{
    bool genreDitemukan = false;

    // Mencocokkan Genre
    for (int j = 0; j < gameListMenu[i].genreCount; j++)
    {
        for (int k = 0; k < selectedGenreCount; k++)
        {
            if (gameListMenu[i].genre[j] == selectedGenres[k])
            {
                genreDitemukan = true;
                break;
            }
        }
        if (genreDitemukan)
            break;
    }

    // Mengeliminasi Game jika tidak memiliki Genre yang dicari
    if (!genreDitemukan)
    {
        for (int l = i; l < MAX_DATA - 1; l++)
        {
            gameListMenu[l] = gameListMenu[l + 1];
        }
        gameListMenuCount--;
    }
    else
        i++;
}

// Sorting Menu
switch (gameSortType)
{
case 1:
    InsertionSortAscending_NamaGame(gameListMenu, gameListMenuCount);

```



```

        break;
    case 2:
        SelectionSortDescending_JumlahModGame(gameListMenu, gameListMenuCount);
        break;
    }

    // Mengosongkan sisa data game
    for (int i = gameListMenuCount; i < sizeof(gameListMenu) / sizeof(Game);
i++)
        gameListMenu[i] = Game();

    // Menampilkan Menu Games
    cout << "Games" << endl;
    Border("-", 106);
    if (currentUser->level == "admin")
    {
        cout << "C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort |
";
    }
    cout << "F > Filter Game" << endl;
    Border("-", 106);
    if (gameListMenuCount > 0)
    {
        cout << left << setw(6) << "No."
            << setw(40) << "Game"
            << setw(10) << "Jumlah Mod " << (gameSortType == 1 ? "(Desc)" : "")
            << endl;
        Border("-", 106);

        for (int i = 0; i < gameListMenuCount; i++)
        {
            cout << left << setw(4) << i + 1 << setw(2) << ">"
                << setw(40) << gameListMenu[i].nama
                << right << setw(10) << gameListMenu[i].jumlahMod
                << endl;
        }
    }
    else
    {
        cout << "Belum ada data game!" << endl;
    }
    Border("-", 106);
    cout << "Q > Keluar | B > Kembali | L > Logout" << endl;
    cout << endl;
    cout << "Pilih Menu\t: ";
    getline(cin, _temp);

    // Mencari pilihan game
    for (int i = 0; i < gameCount; i++)
    {
        if (to_string(i + 1) == _temp)

```

```

        {
            selectedGame = gameListMenu[i];
            break;
        }
    }

    if (selectedGame.nama != "")
    {
        // Pindah ke Menu Mod yang terkait dengan game
        halaman = '2';
        return;
    }

    // List Pilihan
    if (currentUser->level == "admin" && (_temp == "C" || _temp == "c"))
    {
        // Tambah Game
        TambahGame(gamesPtr, gameCount, dataArrayPtr, genreCount);
    }
    else if (currentUser->level == "admin" && _temp == "U" || _temp == "u")
    {
        // Edit Game
        EditGame(gameListMenu, gameListMenuCount, gamesPtr, gameCount,
dataArrayPtr, genreCount);
    }
    else if (currentUser->level == "admin" && _temp == "D" || _temp == "d")
    {
        // Hapus Game
        HapusGame(gameListMenu, gameListMenuCount, gamesPtr, gameCount, modsPtr,
modCount);
    }
    else if (_temp == "S" || _temp == "s")
    {
        // Hapus Game
        SortGame(gameSortType);
    }
    else if (_temp == "F" || _temp == "f")
    {
        // Pindah menu filter
        halaman = '3';
        return;
    }
    else if ((_temp == "Q" || _temp == "q") || (_temp != "B" || _temp != "b") ||
(_temp == "L" || _temp == "l"))
    {
        pilihan_menu = _temp;
    }
    else
    {
        cout << "Menu tidak ditemukan!";
        getline(cin, _temp);
    }
}

```

```
}  
}
```

## J. Menu Filter Game

Menu “Filter Game” menyediakan cara untuk melihat daftar game lebih spesifik sesuai dengan tipe game yang dicari pengguna. Dalam menu ini terdapat salah satunya adalah filter “Genre.”

### Source Code:

```
void MenuFilterGame(char &halaman, string &pilihan_menu, string  
selectedGenres[], int &selectedGenreCount, GabunganArray *dataArrayPtr, int  
&genreCount)  
{  
    string _temp = "";  
  
    cout << "Filter Game" << endl;  
    Border("-", 106);  
    cout << "1. Genre" << endl;  
    Border("-", 106);  
    cout << "Q > Keluar | B > Kembali | L > Logout" << endl;  
    cout << endl;  
    cout << "Pilih Menu\t: ";  
    getline(cin, _temp);  
    Border("-", 106);  
  
    // List Pilihan  
    if (_temp == "1")  
    {  
        // Filter Genre  
        string genres[SMOL_DATA];  
        for (int i = 0; i < genreCount; i++)  
        {  
            genres[i] = dataArrayPtr[i].genre;  
        }  
        PilihanCheckbox(selectedGenres, selectedGenreCount, genres, genreCount,  
"Genre");  
        cout << endl;  
    }  
    else if ((_temp == "Q" || _temp == "q") || (_temp != "B" || _temp != "b") ||  
(_temp == "L" || _temp == "l"))  
    {  
        pilihan_menu = _temp;  
    }  
}
```

```

else
{
    cout << "Menu tidak ditemukan!";
    getline(cin, _temp);
}
}

```

## N. Upload Mod

Fitur “Upload Mod” memungkinkan program untuk menambahkan *mod* baru dari suatu *game* yang telah terdaftar. Dalam proses upload mod, masukan yang diperlukan terdiri dari nama, *video game*, nama *creator*, dan deskripsi dari mod yang di-upload-nya.

### Source Code:

```

void UploadMod(Mod *modsPtr, int &modCount, Game *gamesPtr, int &gameCount, User
*currentUser)
{
    string _temp = "";

    if (modCount < MAX_DATA)
    {
        // Tambahkan mod jika belum penuh
        Mod modBaru;

        Border("-", 106);
        cout << "Upload Mod" << endl;
        Border("-", 106);
        cout << "(Kosongkan input untuk kembali.)" << endl;

        // Input nama mod
        cout << "Nama Mod\t: ";
        getline(cin, modBaru.nama);
        cout << endl;

        // Validasi nama mod
        for (int i = 0; i < modCount; i++)
        {
            if (modBaru.nama != "" && modBaru.nama == modsPtr[i].nama)
            {
                modBaru.nama = "";
                cout << "Nama mod sudah ada!";
                getline(cin, _temp);
                break;
            }
        }

        if (modBaru.nama == "")

```

```

        return;

// Input creator
cout << "Creator\t\t: ";
getline(cin, modBaru.creator);
cout << endl;

if (modBaru.creator == "")
    return;

// Input pilihan game
cout << "Game" << endl;
for (int i = 0; i < gameCount; i++)
    cout << i + 1 << ". " << gamesPtr[i].nama << endl;
cout << "Pilih Game\t: ";
getline(cin, _temp);
cout << endl;

if (_temp == "")
    return;

// Mencari dan validasi game yang dipilih
for (int i = 0; i < gameCount; i++)
{
    if (to_string(i + 1) == _temp)
    {
        modBaru.game = &gamesPtr[i];
        break;
    }
}

if (modBaru.game->nama == "")
{
    cout << "Game tidak ditemukan!";
    getline(cin, _temp);
    return;
}

// Input deskripsi
cout << "Deskripsi\t: ";
getline(cin, modBaru.deskripsi);
cout << endl;

if (modBaru.deskripsi == "")
    return;

// Proses penambahan mod
modBaru.id += 1;
modBaru.uploader = currentUser;
modsPtr[modCount] = modBaru;
modCount++;

```

```

        cout << "Mod berhasil diupload!";
        getline(cin, _temp);
    }
    else
    {
        cout << "Mod penuh! Tidak dapat mengupload mod lagi.";
        getline(cin, _temp);
    }
}

```

## O. Edit Mod

Fitur “Edit Mod” memungkinkan *uploader* untuk memperbarui data *mod* yang sedang dibuka di menu “Detail Mod.” *Uploader* dapat memperbarui nama, nama *creator*, dan deskripsi dari mod yang telah dipilihnya.

### Source Code:

```

void EditMod(Mod &selectedMod, Mod *modsPtr, int modCount)
{
    string _temp = "";

    cout << "(Kosongkan untuk melewati salah satu pengeditan.)" << endl;
    cout << "Nama Baru\t:";
    getline(cin, _temp);

    for (int i = 0; i < modCount; i++)
    {
        if (_temp != "" && modsPtr[i].nama == _temp)
        {
            _temp = "";
            cout << "Nama mod sudah ada!";
            getline(cin, _temp);
            break;
        }
    }

    if (_temp != "")
        selectedMod.nama = _temp;

    cout << "Creator Baru\t:";
    getline(cin, _temp);

    if (_temp != "")
        selectedMod.creator = _temp;

    cout << "Deskripsi Baru\t:";
    getline(cin, _temp);
}

```

```

    if (_temp != "")
        selectedMod.deskripsi = _temp;

    for (int i = 0; i < modCount; i++)
    {
        if (modsPtr[i].id == selectedMod.id)
        {
            modsPtr[i] = selectedMod;
        }
    }

    cout << "Mod berhasil diperbarui!";
    getline(cin, _temp);
}

```

## P. Hapus Mod

Fitur “Edit Mod” memungkinkan *uploader* untuk menghapus *mod* yang sedang dibuka di menu “Detail Mod.”

### Source Code:

```

void HapusMod(char &halaman, Mod &selectedMod, Mod *modsPtr, int modCount)
{
    string _temp = "";
    // Menghapus Mod saat ini
    cout << "Apakah Anda yakin ingin menghapus mod \"" << selectedMod.nama <<
    "\"? [Y/N] ";
    getline(cin, _temp);

    if (_temp == "")
        return;

    if (_temp == "Y" || _temp == "y")
    {
        // Proses penghapusan mod
        for (int i = 0; i < modCount; i++)
        {
            if (modsPtr[i].id == selectedMod.id)
            {
                for (int j = i; j < modCount; j++)
                {
                    modsPtr[j] = modsPtr[j + 1];
                }
                modCount--;
                break;
            }
        }
    }
}

```



```

    }
    cout << "Mod berhasil dihapuskan!";
    getline(cin, _temp);
    selectedMod = Mod();
    halaman = '2';
}
else if (_temp == "N" || _temp == "n")
{
    // Batalkan penghapusan mod
    cout << "Mod batal dihapuskan!";
    getline(cin, _temp);
}
else
{
    cout << "Input tidak valid!";
    getline(cin, _temp);
}
}

```

### Q. Menu Sort Mod

Fitur “Sort Mod”, memungkinkan pengguna dapat memilih tipe sorting data yang ingin ditampilkan dalam daftar mod.

#### Source Code:

```

void SortMod(int &modSortType)
{
    string _temp = "";

    cout << endl;
    cout << "0 > Matikan Sorting" << endl;
    cout << "1 > Nama Mod (Ascending)" << endl;
    cout << "2 > Nama Mod (Descending)" << endl;
    cout << endl;
    cout << "Pilih Tipe Sorting:";
    getline(cin, _temp);

    if (_temp == "")
        return;

    if (_temp == "0")
        modSortType = 0;
    else if (_temp == "1")
        modSortType = 1;
    else if (_temp == "2")
        modSortType = 2;
    else

```

```

    cout << "Pilihan tipe sorting valid!" << endl;
}

```

## R. Menu Mod

Di “Menu Mod”, program menampilkan daftar mod yang tersedia. Program juga menyediakan pengguna fitur untuk meng-*upload* suatu mod..

### Source Code:

```

void MenuMod(char &halaman, string &pilihan_menu, Game selectedGame, Mod
&selectedMod, Game *gamesPtr, int &gameCount, Mod *modsPtr, int &modCount, User
*currentUser, int &modSortType)
{
    string _temp = "";
    Mod modListMenu[MAX_DATA];
    int modListMenuCount = 0;

    selectedMod = Mod();

    // Memfilter mod berdasarkan game yang dipilih
    for (int i = 0; i < modCount; i++)
    {
        if (selectedGame.nama != "" && modsPtr[i].game->nama !=
selectedGame.nama)
            continue;

        modListMenu[modListMenuCount] = modsPtr[i];
        modListMenuCount++;
    }

    switch (modSortType)
    {
    case 1:
        BubbleSort_NamaMod(modListMenu, modCount, false);
        break;
    case 2:
        BubbleSort_NamaMod(modListMenu, modCount, true);
        break;
    }

    // Mengosongkan sisa data mod
    for (int i = modListMenuCount; i < sizeof(modListMenu) / sizeof(Mod); i++)
        modListMenu[i] = Mod();

    // Menampilkan Menu Mod
    if (selectedGame.nama != "")
        cout << "Mod " << selectedGame.nama << endl;
}

```

```

else
    cout << "Semua Mod" << endl;
Border("-", 106);
cout << "C > Upload Mod | S > Sort" << endl;
Border("-", 106);
if (modListMenuCount > 0)
{
    // Daftar Mod
    cout << left << setw(6) << "No."
        << setw(40) << "Nama Mod"
        << setw(40) << "Game"
        << setw(20) << "Creator"
        << endl;
    Border("-", 106);

    for (int i = 0; i < modListMenuCount; i++)
    {
        cout << left << setw(4) << i + 1 << setw(2) << ">"
            << setw(40) << modListMenu[i].nama
            << setw(40) << modListMenu[i].game->nama
            << setw(20) << modListMenu[i].creator
            << endl;
    }
}
else
{
    cout << "Belum ada data mod!" << endl;
}
Border("-", 106);
cout << "Q > Keluar | B > Kembali | L > Logout" << endl;
cout << endl;
cout << "Pilih Menu\t: ";
getline(cin, _temp);

// Pilih Mod
for (int i = 0; i < modListMenuCount; i++)
{
    if (to_string(i + 1) == _temp)
    {
        selectedMod = modListMenu[i];
    }
}

if (selectedMod.id > 0)
{
    halaman = '4';
    return;
}

// List Pilihan
if (_temp == "C" || _temp == "c")

```

```

{
    // UpLoad Mod
    UploadMod(modsPtr, modCount, gamesPtr, gameCount, currentUser);
}
if (_temp == "S" || _temp == "s")
{
    // Sort Mod
    SortMod(modSortType);
}
else if ((_temp == "Q" || _temp == "q") || (_temp != "B" || _temp != "b") ||
(_temp == "L" || _temp == "l"))
{
    pilihan_menu = _temp;
}
else
{
    cout << "Menu tidak ditemukan!";
    getline(cin, _temp);
}
}

```

## S. Detail Mod

Menu “Detail Mod” menampilkan informasi lengkap dari suatu *mod* yang dipilih dari “Menu Mod.” Dalam menu ini menyediakan fitur bagi *uploader* untuk mengedit atau menghapus mod yang sedang dilihatnya saat ini.

### Source Code:

```

void DetailMod(char &halaman, string &pilihan_menu, Mod &selectedMod, Mod
*modsPtr, int &modCount, User *currentUser)
{
    string _temp = "";

    if (selectedMod.id < 1)
    {
        halaman = '2';
        return;
    }

    // Menampilkan detail mod terpilih
    cout << selectedMod.nama << endl;
    Border("-", 106);
    if (currentUser->id == selectedMod.uploader->id)
    {
        cout << "U > Edit | D > Hapus" << endl;
        Border("-", 106);
    }
}

```

```

else if (currentUser->level == "admin")
{
    cout << "D > Hapus" << endl;
    Border("-", 106);
}
cout << "Creator\t\t: " << selectedMod.creator << endl;
cout << "Uploader\t: " << selectedMod.uploader->username << endl;
cout << "Game\t\t: " << selectedMod.game->nama << endl;
Border("-", 106);
cout << selectedMod.deskripsi << endl;
Border("-", 106);
cout << "Q > Keluar | B > Kembali | L > Logout" << endl;
cout << endl;
cout << "Pilih Menu\t: ";
getline(cin, _temp);
Border("-", 106);

if (currentUser->id == selectedMod.uploader->id && (_temp == "U" || _temp ==
"u"))
{
    // Edit Mod terpilih
    EditMod(selectedMod, modsPtr, modCount);
}
else if ((currentUser->id == selectedMod.uploader->id || currentUser->level
== "admin") && (_temp == "D" || _temp == "d"))
{
    // Hapus Mod terpilih
    HapusMod(halaman, selectedMod, modsPtr, modCount);
}
else if ((_temp == "Q" || _temp == "q") || (_temp != "B" || _temp != "b") ||
(_temp == "L" || _temp == "l"))
{
    pilihan_menu = _temp;
}
else
{
    cout << "Menu tidak ditemukan!";
    getline(cin, _temp);
}
}

```

## 4. Hasil Output

```
=====
Sistem Pengelolaan Mod
=====
1 > Login
2 > Register
=====
Q > Keluar

Pilih Menu      : 
```

Gambar 4.1 Memulai Program dan masuk di Sistem Login

```
Username       : Hermawan
Password       : 2409106060
Login berhasil!
=====
```

Gambar 4.2 Proses Login Berhasil

```
=====
Menu Utama
=====
Selamat datang, Hermawan!

1 > Games
2 > Mods
=====
Q > Keluar | L > Logout

Pilih Menu      : 
```

Gambar 4.3 Halaman Menu Utama

```
=====
Games
=====
C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort | F > Filter Game
=====


| No. | Game             | Jumlah Mod |
|-----|------------------|------------|
| 1   | > Stardew Valley | 1          |
| 2   | > Undertale      | 1          |
| 3   | > Terraria       | 2          |
| 4   | > Fallout 4      | 0          |
| 5   | > Skyrim         | 0          |


=====
Q > Keluar | B > Kembali | L > Logout

Pilih Menu      : 
```

Gambar 4.4 Halaman Menu Game yang belum terurut

```

Tipe Sorting:
0 > Matikan Sorting
1 > Nama Game (Ascending)
2 > Jumlah Game (Descending)

Pilih Tipe Sorting:

```

Gambar 4.5 Fitur memilih tipe sorting daftar Menu Game

```

=====
Games
-----
C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort | F > Filter Game
-----

```

No.	Game (Asc)	Jumlah Mod
1	> Fallout 4	0
2	> Skyrim	0
3	> Stardew Valley	1
4	> Terraria	2
5	> Undertale	1

```

-----
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      : 

```

Gambar 4.6 Halaman Menu Game dengan tipe sorting “Nama Game (Ascending)”

```

Tipe Sorting:
0 > Matikan Sorting
1 > Nama Game (Ascending)
2 > Jumlah Game (Descending)

Pilih Tipe Sorting:

```

Gambar 4.7 Memilih fitur tipe sorting untuk mengganti tipe sorting daftar game



```
=====
```

Games		
C > Tambah Game   U > Edit Game   D > Hapus Game   S > Sort   F > Filter Game		
No.		Jumlah Mod (Desc)
1	> Fallout 4	0
2	> Skyrim	0
3	> Stardew Valley	1
4	> Undertale	1
5	> Terraria	2
Q > Keluar   B > Kembali   L > Logout		
Pilih Menu : <input type="text"/>		

Gambar 4.8 Halaman Menu Game dengan tipe sorting “Jumlah Game (Descending)”

```
=====
```

Tambah Game	
Nama Game	: Geometry Dash
Genre	
1. Action	
2. Adventure	
3. RPG	
4. Simulation	
(Kosongkan input untuk mengakhiri pilihan.)	
Pilih pilihan:	1
Genre	
1. Action (Dipilih)	
2. Adventure	
3. RPG	
4. Simulation	
(Kosongkan input untuk mengakhiri pilihan.)	
Pilih pilihan:	
Game berhasil ditambahkan!	

```
=====
```

Gambar 4.9 Menambahkan Satu Game Baru

```
=====
Games
-----
C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort | F > Filter Game
-----
No.                                     Jumlah Mod (Desc)
-----
1 > Fallout 4                           0
2 > Skyrim                              0
3 > Geometry Dash                        0
4 > Stardew Valley                        1
5 > Undertale                            1
6 > Terraria                             2
-----
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      : █
```

Gambar 4.10 Halaman Menu Game setelah menambahkan “Geometry Dash” dengan data yang masih tersortir berdasarkan “Jumlah Mod (Descending)”

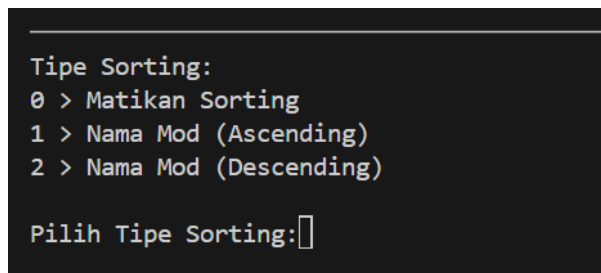
```
=====
Menu Utama
-----
Selamat datang, Hermawan!

1 > Games
2 > Mods
-----
Q > Keluar | L > Logout
Pilih Menu      : █
```

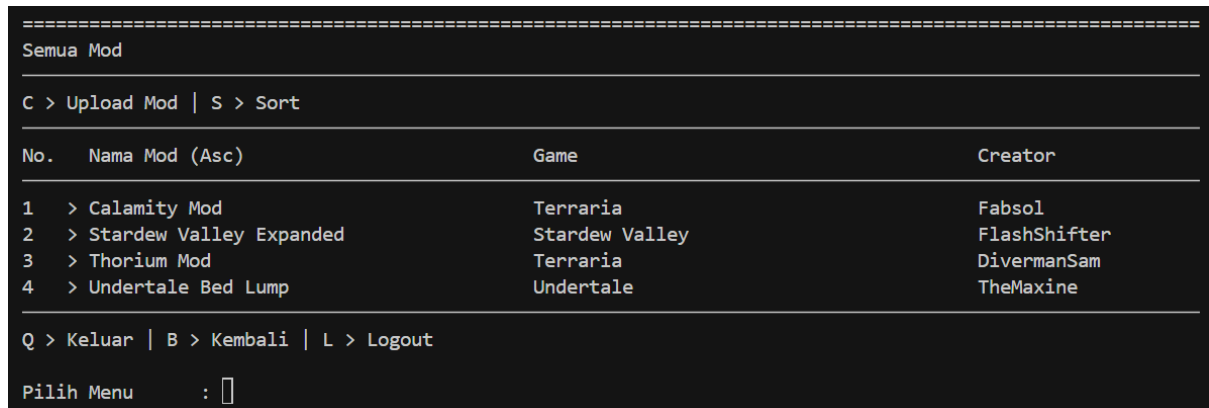
Gambar 4.11 Kembali ke Menu Utama

```
=====
Semua Mod
-----
C > Upload Mod | S > Sort
-----
No.      Game      Creator
-----
1 > Stardew Valley Expanded    Stardew Valley    FlashShifter
2 > Undertale Bed Lump         Undertale          TheMaxine
3 > Calamity Mod               Terraria          Fabsol
4 > Thorium Mod                Terraria          DivermanSam
-----
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      : █
```

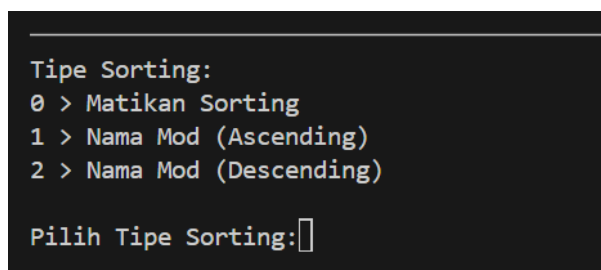
Gambar 4.12 Halaman Menu Mod dengan daftar mod yang belum tersortir.



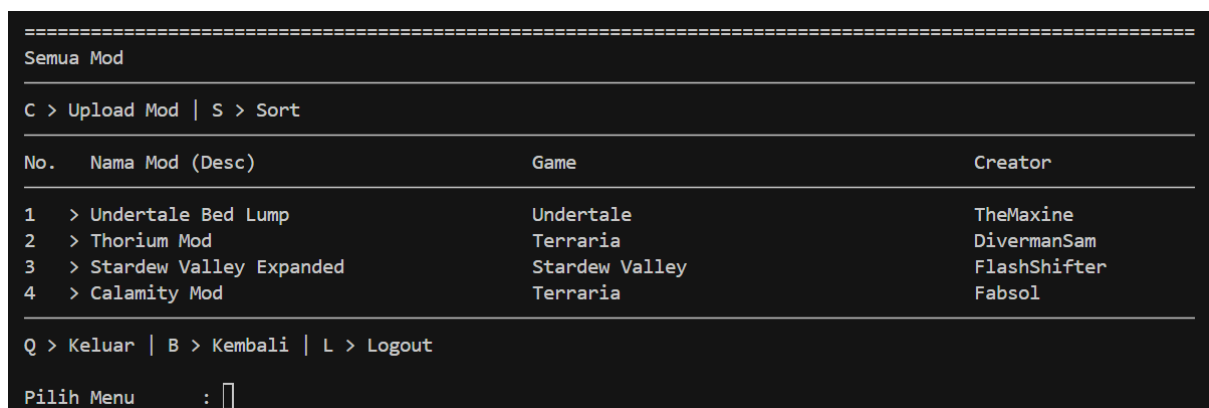
Gambar 4.13 Fitur memilih tipe sorting untuk Menu Mod



Gambar 4.14 Halaman Menu Mod dengan daftar mod yang disortir berdasarkan “Nama Mod (Ascending)”



Gambar 4.15 Memilih ulang tipe sorting di Menu Mod



Gambar 4.16 Tampilan “Menu Game” setelah memilih filter genre “Action”

```

Upload Mod

(Kosongkan input untuk kembali.)
Nama Mod      : Globed

Creator       : dank_meme

Game
1. Stardew Valley
2. Undertale
3. Terraria
4. Fallout 4
5. Skyrim
6. Geometry Dash
Pilih Game    : 6

Deskripsi     : Globed is an unofficial third-party modification for Geometry Dash that adds multiplayer capabilities to the game, allowing you
to see other players and play levels with them.

Mod berhasil diupload!
=====

```

Gambar 4.17 Menambahkan satu mod baru untuk game “Geometry Dash”

```

=====
Semua Mod

C > Upload Mod | S > Sort

No.   Nama Mod (Desc)                Game                Creator
-----
1 > Undertale Bed Lump              Undertale            TheMaxine
2 > Thorium Mod                     Terraria            DivermanSam
3 > Stardew Valley Expanded          Stardew Valley      FlashShifter
4 > Globed                          Geometry Dash        dank_meme
5 > Calamity Mod                    Terraria            Fabsol

Q > Keluar | B > Kembali | L > Logout

Pilih Menu    : 

```

Gambar 4.18 Halaman Menu Mod setelah menambahkan mod “Globed” dengan daftar mod yang masih tersortir berdasarkan “Nama Mod (Descending)”

```

Tipe Sorting:
0 > Matikan Sorting
1 > Nama Mod (Ascending)
2 > Nama Mod (Descending)

Pilih Tipe Sorting:

```

Gambar 4.19 Memilih ulang tipe sorting di Menu Mod

```

=====
Semua Mod
=====
C > Upload Mod | S > Sort
=====
No.                                     Game                                     Creator
=====
1  > Stardew Valley Expanded           Stardew Valley                           FlashShifter
2  > Undertale Bed Lump                 Undertale                                TheMaxine
3  > Calamity Mod                       Terraria                                 Fabsol
4  > Thorium Mod                       Terraria                                 DivermanSam
5  > Globed                             Geometry Dash                             dank_meme
=====
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      : 

```

Gambar 4.20 Halaman Menu Mod setelah mematikan sortir

```

=====
Menu Utama
=====
Selamat datang, Hermawan!

1 > Games
2 > Mods
=====
Q > Keluar | L > Logout
Pilih Menu      : 

```

Gambar 4.21 Kembali ke halaman Menu Utama

```

=====
Games
=====
C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort | F > Filter Game
=====
No.                                     Jumlah Mod (Desc)
=====
1  > Fallout 4                           0
2  > Skyrim                             0
3  > Stardew Valley                       1
4  > Undertale                           1
5  > Geometry Dash                       1
6  > Terraria                           2
=====
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      : 

```

Gambar 4.22 Halaman Menu Game setelah menambah mod untuk game “Geometry Dash”

```

Tipe Sorting:
0 > Matikan Sorting
1 > Nama Game (Ascending)
2 > Jumlah Game (Descending)

Pilih Tipe Sorting:

```

Gambar 4.23 Memilih ulang tipe sorting di Menu Game

```

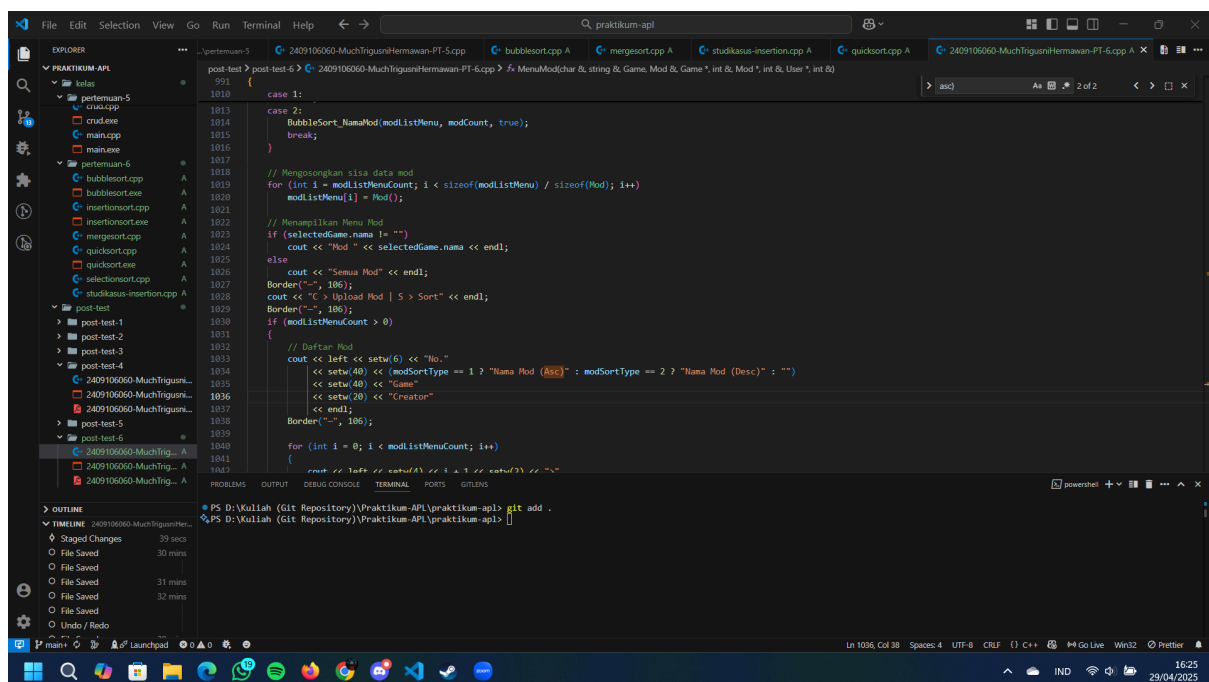
=====
Games
=====
C > Tambah Game | U > Edit Game | D > Hapus Game | S > Sort | F > Filter Game
=====
No.                                Jumlah Mod
=====
1 > Stardew Valley                  1
2 > Undertale                      1
3 > Terraria                       2
4 > Fallout 4                      0
5 > Skyrim                        0
6 > Geometry Dash                  1
=====
Q > Keluar | B > Kembali | L > Logout
Pilih Menu      :

```

Gambar 4.24 Halaman Menu Game dengan daftar game tanpa tipe sorting

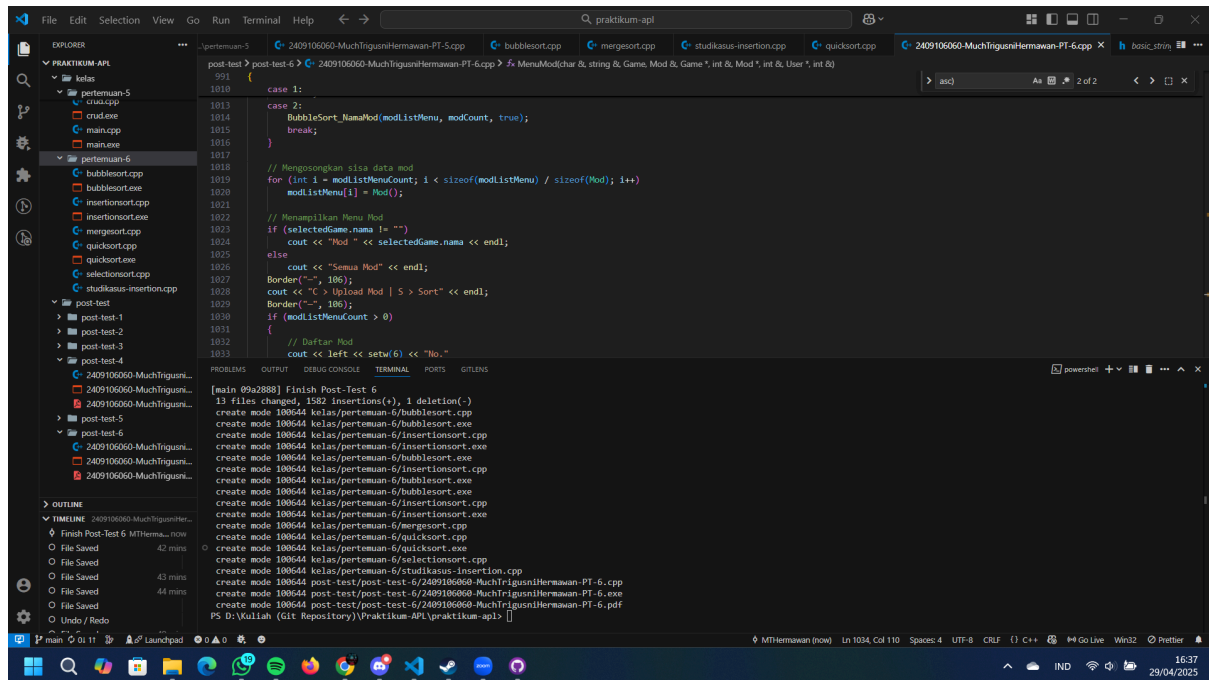
## 5. Langkah-Langkah Git

### A. Git Add



Menambahkan semua perubahan file saat ini ke commit berikutnya dengan “git add .”

## B. Git Commit

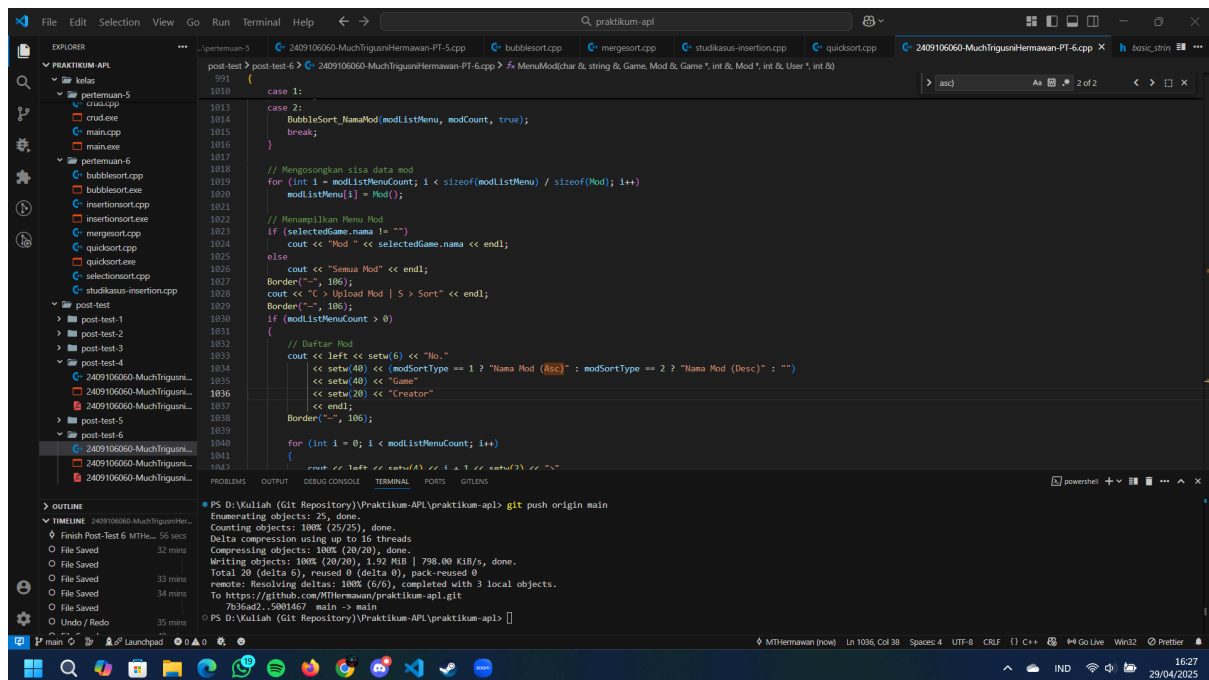


```
git commit -m "Finish Post-Test 6"
```

13 files changed, 1582 insertions(+), 1 deletion(-)  
create mode 108644 kelas/pertemuan-6/bubbleSort.exe  
create mode 108644 kelas/pertemuan-6/insertionSort.exe  
create mode 108644 kelas/pertemuan-6/mergeSort.exe  
create mode 108644 kelas/pertemuan-6/quickSort.exe  
create mode 108644 kelas/pertemuan-6/selectionSort.exe  
create mode 108644 kelas/pertemuan-6/studKasus-insertion.cpp  
create mode 108644 kelas/pertemuan-6/bubbleSort.exe  
create mode 108644 kelas/pertemuan-6/insertionSort.exe  
create mode 108644 kelas/pertemuan-6/mergeSort.exe  
create mode 108644 kelas/pertemuan-6/quickSort.exe  
create mode 108644 kelas/pertemuan-6/selectionSort.exe  
create mode 108644 kelas/pertemuan-6/studKasus-insertion.cpp  
create mode 108644 post-test/post-test-6/2409106060-MuchTrigunihHermawan-PT-6.cpp  
create mode 108644 post-test/post-test-6/2409106060-MuchTrigunihHermawan-PT-6.pdf  
PS D:\Kuliah (git Repository)\Praktikum-API\praktikum-api> |

Membuat checkpoint progres pada branch main dengan git commit dengan pesan “Finish Post-Test-6”.

## C. Git Push



```
git push origin main
```

Enumerating objects: 25, done.  
Counting objects: 100% (25/25), done.  
Delta compression using up to 16 threads  
Compressing objects: 100% (20/20), done.  
Writing objects: 100% (20/20), 1.92 MiB | 796.00 KiB/s, done.  
Total 20 (delta 6), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (6/6), completed with 3 local objects.  
To https://github.com/MTHerawan/praktikum-api.git  
7b36ad2..5801467 main -> main  
PS D:\Kuliah (git Repository)\Praktikum-API\praktikum-api> |

Dengan melakukan git push, perubahan file saat ini telah disinkronisasikan di Repository GitHub.