

LAPORAN PRAKTIKUM
POSTTEST 6
ALGORITMA PEMROGRAMAN LANJUT

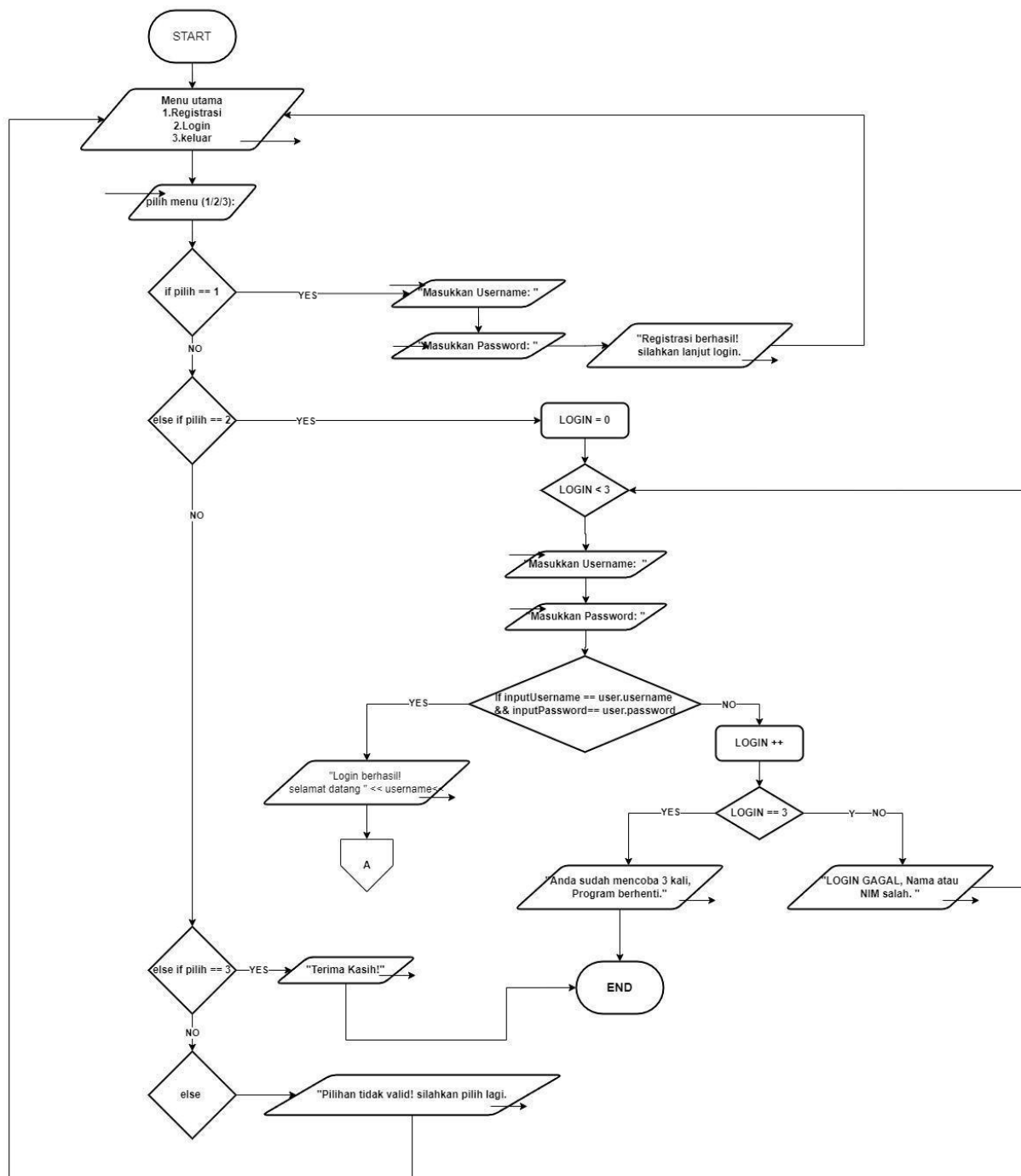


Disusun oleh:
Nadia Rahmah (2409106018)
Kelas (A1 '24)

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

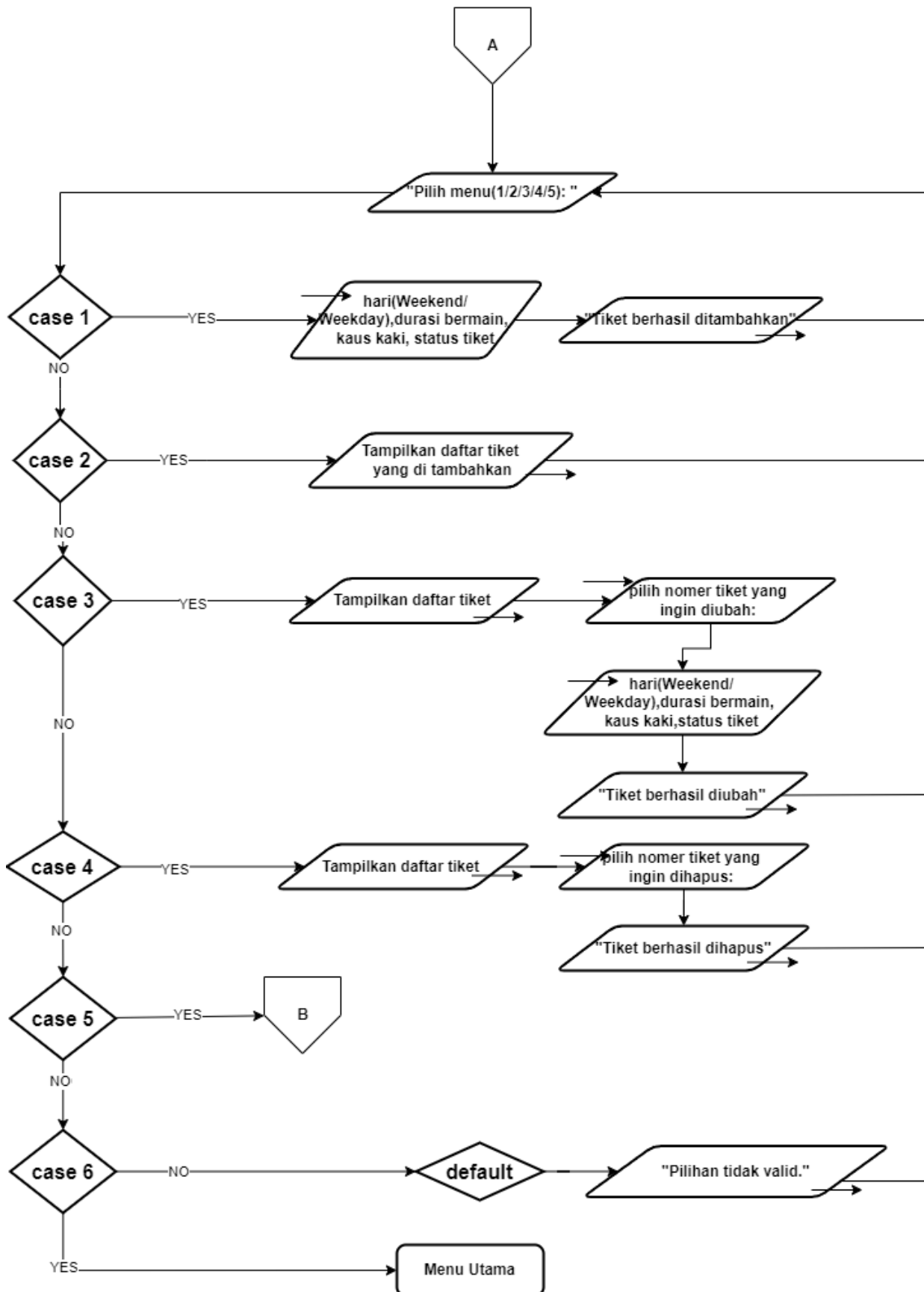
1. Flowchart

MENU UTAMA



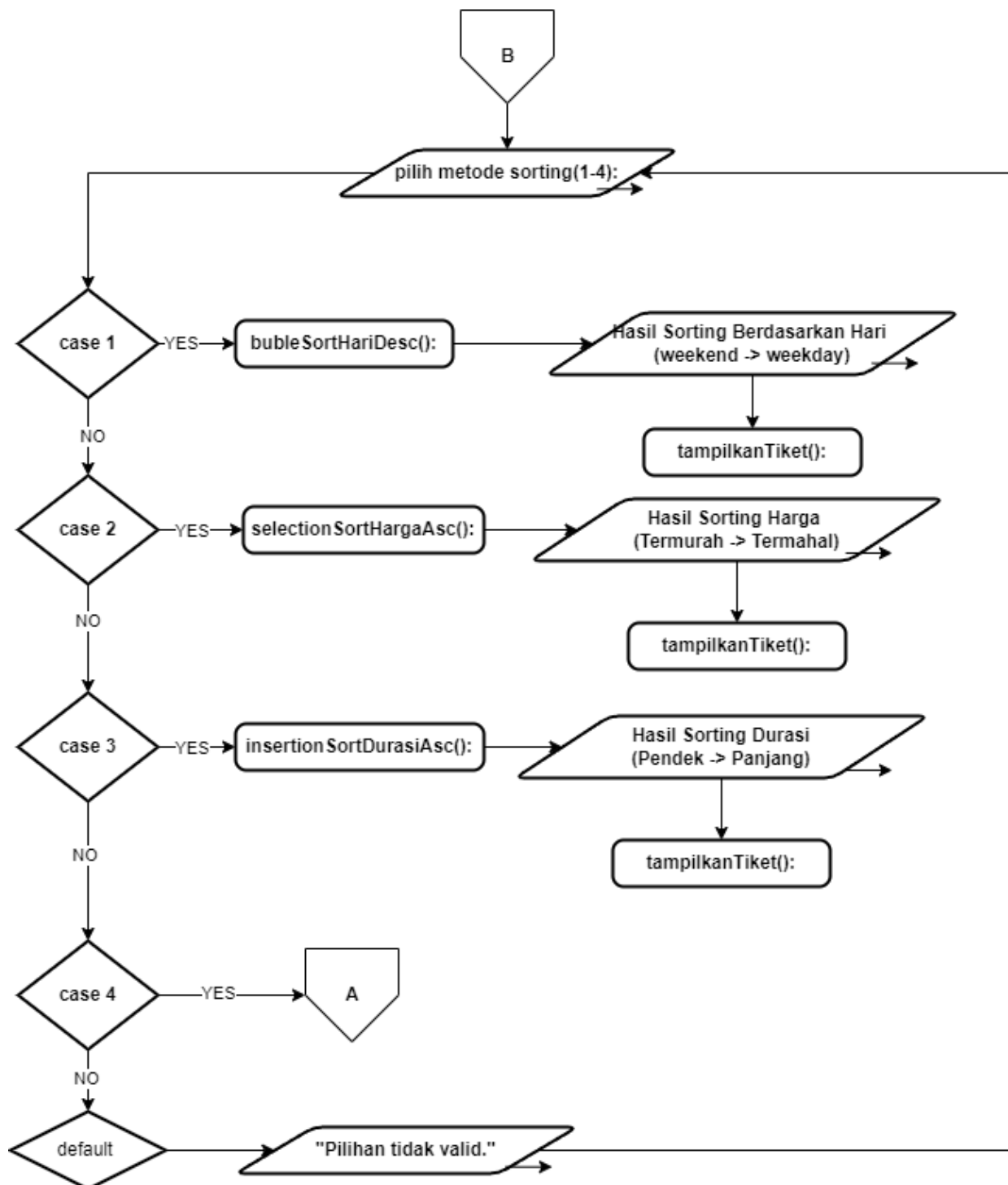
Gambar 1.1 Flowchart Menu Utama

MENU TIKET



Gambar 1.2 Flowchart Menu Tiket

Menu Sorting



Gambar 1.1 Flowchart Menu Sorting

2. Analisis Program

2.1 Deskripsi Singkat Program

Program ini dibuat menggunakan Bahasa pemrograman C++ dengan judul “Manajemen pembelian tiket *Mini Playground*” untuk memudahkan pengelolaan data tiket. Struktur data pada program ini menggunakan struct Tiket, struct User, array tiket, array user dan dilengkapi array of struct (Tiket daftarTiket[MaxTiket]. Masing-masing fitur yang ada pada menu program dipisahkan dari program utama menjadi subprogram menggunakan fungsi dan prosedur. Fitur tersebut diantaranya Registrasi, Login, Tambah tiket, Tampilkan tiket, Ubah tiket, Hapus tiket dan menu Sorting. Program ini telah menerapkan prinsip pointer dengan menggunakan parameter address-of(&) pada pemanggilan fungsi registrasi dan login serta menggunakan parameter deference(*) pada fungsi registrasi dan login. Pengaplikasian algoritma sorting seperti Buble sort, Selection sert, dan Insertion sort untuk memudahkan pengurutan secara ascending dan descending. Pada program juga dilengkapi proses keamanan login untuk memastikan hanya pengguna yang sudah ditetapkan yang bisa mengakses program dan juga membatasi jumlah percobaan login sebanyak 3 kali.

3. Source Code

A. Menu Utama

Pada menu utama terdapat tiga pilihan menu yaitu menu Registrasi, menu login, dan menu keluar. Terdapat fungsi Registrasi dan fungsi Login, Program akan memanggil fungsi yang sesuai dengan pilihan user. Menu keluar untuk mengakhiri program karena program ini menggunakan loop yang membuat program akan terus berulang.

Source Code:

```
void menuUtama(){
    User user;
    while (true){
        cout << "=====" << endl;
        cout << "|      MENU UTAMA      |" << endl;
        cout << "=====" << endl;
        cout << "1. Registrasi\n";
        cout << "2. Login\n";
        cout << "3. Keluar\n";
        cout << "Pilih Menu (1/2/3): ";
        int pilih;
        cin >> pilih;

        if (pilih == 1){
            registrasi(&user);
        }else if(pilih == 2){
```

```

        if (login(&user)){
            menuTiket();
        }
    } else if (pilih == 3) {
        cout << "Terima Kasih!" << endl;
        break;
    } else {
        cout << "Pilihan tidak valid! Silakan pilih lagi." << endl;
    }
}
}
}

```

B. Fitur Registrasi

Fitur Registrasi digunakan untuk mendaftarkan akun baru dengan membuat username dan password. Data akun user baru akan disimpan dalam array of struct sehingga program ini dapat multiuser. Registrasi menerapkan fungsi yang menggunakan parameter pointer(`User *user`) .

Source Code:

```

void registrasi(User *user) {
    cout << "=== Registrasi ===" << endl;
    cout << "Masukkan Username: ";
    cin.ignore();
    getline(cin, daftarUser[jumlahUser].username);
    cout << "Masukkan Password: ";
    getline(cin, daftarUser[jumlahUser].password);

    *user = daftarUser[jumlahUser];

    cout << "Registrasi berhasil! Silakan lanjut login." << endl;
    jumlahUser++;
}

```

C. Fitur Login

Fitur Login digunakan untuk mengamankan akses program, memastikan hanya pengguna yang sudah memiliki akun yang dapat login. Fitur ini juga membatasi percobaan sebanyak 3 kali login untuk memberi kesempatan untuk memasukkan *input*-an yang benar. Fitur Login juga menerapkan fungsi menggunakan parameter pointer(`User *user`)..

Source Code:

```

bool login(User *user) {
    string inputUsername, inputPassword;
    int login = 0;
    while (login < 3) {
        cout << "=== Login ===" << endl;
        cout << "Masukkan Username: ";
        cin >> ws;
        getline(cin, inputUsername);
        cout << "Masukkan Password: ";
        cin >> ws;
        getline(cin, inputPassword);

        for (int i = 0; i < jumlahUser; i++) {
            if (inputUsername == daftarUser[i].username && inputPassword ==
daftarUser[i].password) {
                *user = daftarUser[i];
                cout << "Login berhasil! Selamat datang " << user->username <<
"!" << endl;
                return true;
            }
        }
        login++;
        if (login < 3) {
            cout << "Username atau password salah! Coba lagi" << endl;
        }
    }
    cout << "Anda sudah mencoba 3 Kali. Program berhenti." << endl;
    exit(0);
}

```

D. Menu CRUD Tiket

Fitur ini memiliki beberapa menu memudahkan dalam pengelolaan tiket. Menu dapat diakses dengan melakukan pemanggilan fungsi.

Source Code:

```

void menuTiket() {
    while (true) {
        cout << "======" << endl;
        cout << "|          MENU TIKET          |" << endl;
        cout << "======" << endl;
        cout << "1. Tambah Tiket" << endl;
        cout << "2. Tampilkan Tiket" << endl;
        cout << "3. Ubah data Tiket" << endl;
        cout << "4. Hapus Tiket" << endl;
    }
}

```

```

    cout << "5. Menu Sorting tiket" << endl;
    cout << "6. Keluar ke menu utama" << endl;
    cout << "Pilih menu (1/2/3/4/5): ";
    int pilihan;
    cin >> pilihan;
    switch (pilihan) {
        case 1:
            tambahTiket();
            break;
        case 2:
            tampilkanTiket();
            break;
        case 3:
            ubahTiket();
            break;
        case 4:
            hapusTiket();
            break;
        case 5:
            menuSorting();
            break;
        case 6:
            return;
        default:
            cout << "Pilihan tidak valid" << endl;
    }
}
}

```

E. Fitur Tambah Tiket

Fitur ini digunakan untuk menambahkan data baru ke dalam Array. Data ditambahkan dengan input hari(Weekend/Weekday), durasi bermain, apakah membawa Kaus kaki(Y/T) dan menghitung total harga tiket berdasarkan data tersebut. Data status tiket juga ditambahkan dengan bersifat default “Aktif”.

Source Code:


```

void tambahTiket() {
    cout << "=== MENU TAMBAH TIKET ===" << endl;
    if (jumlahTiket < MaxTiket) {
        Tiket tiketBaru;
        int inputHari, inputDurasi;
        char inputKausKaki;

        cout << "1. Weekend (+ 5.000)" << endl;
        cout << "2. Weekday" << endl;
        cout << "Masukkan Hari (1/2): ";
        cin >> inputHari;
        tiketBaru.hari = (inputHari == 1) ? "Weekend" : "Weekday";

        cout << "1. 1 Jam 20.000" << endl;
        cout << "2. 2 Jam 35.000" << endl;
        cout << "3. Sepuasnya 45.000" << endl;
        cout << "Masukkan Durasi (1/2/3): ";
        cin >> inputDurasi;
        if (inputDurasi == 1) {
            tiketBaru.durasi = "1 Jam";
            tiketBaru.harga = 20000;
        } else if (inputDurasi == 2) {
            tiketBaru.durasi = "2 Jam";
            tiketBaru.harga = 35000;
        } else {
            tiketBaru.durasi = "Sepuasnya";
            tiketBaru.harga = 45000;
        }

        cout << "***Jika tidak membawa kaus kaki, maka diwajibkan membeli
seharga 10.000 ***" << endl;
        cout << "Apakah Anda membawa kaus kaki? (Y untuk Ya, T untuk Tidak):
";
        cin >> inputKausKaki;
        tiketBaru.kausKaki = (inputKausKaki == 'Y' || inputKausKaki == 'y');
        tiketBaru.totalHarga = tiketBaru.harga + ((inputHari == 1) ? 5000 :
0) + (tiketBaru.kausKaki ? 0 : 10000);
        tiketBaru.status = "Aktif";
        daftarTiket[jumlahTiket++] = tiketBaru;
        cout << "Tiket berhasil ditambahkan!" << endl;
    } else {
        cout << "Jumlah tiket sudah mencapai batas maksimum!" << endl;
    }
}

```

F. Fitur Tampilkan Tiket

Fitur ini digunakan untuk menampilkan semua daftar tiket yang sudah ditambahkan.

Source Code:

```
void tampilkanTiket() {
    cout << "=== MENU TAMPILKAN TIKET ===" << endl;
    if (jumlahTiket == 0) {
        cout << "Tidak ada tiket tersedia" << endl;
    } else {
        cout << setw(10) << "No" << setw(15) << "Hari" << setw(15) <<
        "Durasi" << setw(10) << "Harga" << setw(15) << "Kaus Kaki" << setw(15) <<
        "Total Harga" << setw(15) << "Status" << endl;
        for (int i = 0; i < jumlahTiket; i++) {
            cout << setw(10) << (i + 1)
                << setw(15) << daftarTiket[i].hari
                << setw(15) << daftarTiket[i].durasi
                << setw(10) << daftarTiket[i].harga
                << setw(15) << (daftarTiket[i].kausKaki ? "Ya" : "Tidak")
                << setw(15) << daftarTiket[i].totalHarga
                << setw(15) << daftarTiket[i].status << endl;
        }
    }
}
```

G. Fitur Ubah data Tiket

Fitur ini digunakan untuk mengubah data tiket yang sudah ada, dengan menampilkan daftar tiket terlebih dahulu agar memudahkan user memilih nomor tiket yang ingin diubah. Setelah itu mengubah pilihan hari, durasi, kaus kaki, status tiket dan total harga.

Source Code:

```
void ubahTiket() {
    cout << "=== MENU UBAH TIKET ===" << endl;
    tampilkanTiket();
    int no;
    cout << "Masukkan nomor tiket yang ingin diubah: ";
    cin >> no;

    if (no < 1 || no > jumlahTiket) {
        cout << "Nomor tiket tidak valid!" << endl;
        return;
    }
}
```

```

Tiket &tiket = daftarTiket[no - 1];
int inputHari, inputDurasi;
char inputKausKaki;

cout << "1. Weekend" << endl;
cout << "2. Weekday" << endl;
cout << "Masukkan Hari baru (1/2): ";
cin >> inputHari;
tiket.hari = (inputHari == 1) ? "Weekend" : "Weekday";

cout << "1. 1 Jam 20.000" << endl;
cout << "2. 2 Jam 35.000" << endl;
cout << "3. Sepuasnya 45.000" << endl;
cout << "Masukkan Durasi baru (1/2/3): ";
cin >> inputDurasi;
if (inputDurasi == 1) {
    tiket.durasi = "1 Jam";
    tiket.harga = 20000;
} else if (inputDurasi == 2) {
    tiket.durasi = "2 Jam";
    tiket.harga = 35000;
} else {
    tiket.durasi = "Sepuasnya";
    tiket.harga = 45000;
}

cout << "Apakah Anda membawa kaus kaki? (Y untuk Ya, T untuk Tidak): ";
cin >> inputKausKaki;
tiket.kausKaki = (inputKausKaki == 'Y' || inputKausKaki == 'y');
tiket.totalHarga = tiket.harga + ((inputHari == 1) ? 5000 : 0) +
(tiket.kausKaki ? 0 : 10000);

cout << "Masukkan status baru tiket: ";
cin.ignore();
getline(cin, tiket.status);

cout << "Tiket berhasil diubah!" << endl;
}

```

H. Fitur Hapus Tiket

Fitur ini berfungsi untuk menghapus Tiket dengan menampilkan daftar tiket terlebih dahulu untuk memudahkan user memilih menghapus tiket berdasarkan input nomer tiket. Setelah penghapusan, Tiket yang tersisa akan digeser.

Source Code:

```
void hapusTiket() {
    cout << "=== MENU HAPUS TIKET ===" << endl;
    tampilkanTiket();
    int no;
    cout << "Masukkan nomor tiket yang ingin dihapus: ";
    cin >> no;

    if (no < 1 || no > jumlahTiket) {
        cout << "Nomor tiket tidak valid!" << endl;
        return;
    }

    for (int i = no - 1; i < jumlahTiket - 1; i++) {
        daftarTiket[i] = daftarTiket[i + 1]; // Menggeser tiket ke kiri
    }
    jumlahTiket--; // Mengurangi jumlah tiket
    cout << "Tiket berhasil dihapus!" << endl;
}
```

I. Fitur Shorting

Fitur ini berfungsi untuk mengurutkan data tiket menggunakan algoritma sorting secara Ascending atau descending. Terdapat empat menu diantaranya yaitu Buble sort untuk mengurutkan Hari berdasarkan Descending, Selection sort untuk mengurutkan Harga berdasarkan Ascending, Insertion sort untuk mengurutkan Durasi berdasarkan Ascending, dan menu kembali ke menu tiket.

Source Code:

```
void menuSorting() {
    while (true) {
        cout << "\n===== " << endl;
        cout << "|      MENU SORTING      |" << endl;
        cout << "===== " << endl;
        cout << "1. Sorting berdasarkan Hari (Descending) - Bubble Sort" << endl;
        cout << "2. Sorting Harga (Ascending) - Selection Sort" << endl;
        cout << "3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort" <<
endl;
        cout << "4. keluar ke Menu Tiket" << endl;
        cout << "Pilih metode sorting (1-4): ";

        int pilihan;
        cin >> pilihan;
    }
}
```

```

        switch (pilihan) {
            case 1:
                bubbleSortHariDesc(daftarTiket, jumlahTiket);
                cout << "\nHasil Sorting Berdasarkan Hari (Weekend ->
Weekday)\n";
                tampilkanTiket();
                break;
            case 2:
                selectionSortHargaAsc(daftarTiket, jumlahTiket);
                cout << "\nHasil Sorting Berdasarkan Harga (Termurah ->
Termahal)\n";
                tampilkanTiket();
                break;
            case 3:
                insertionSortDurasiAsc(daftarTiket, jumlahTiket);
                cout << "\nHasil Sorting Berdasarkan Durasi (Pendek ->
Panjang) \n";
                tampilkanTiket();
                break;
            case 4:
                return;
            default:
                cout << "Pilihan tidak valid!\n";
        }
    }
}

```

J. Fitur Buble Sort

Fitur ini berfungsi untuk memudahkan pengurutan Hari berdasarkan Descending, dengan menggunakan algoritma bubble sort untuk melihat tiket Weekend terlebih dahulu.

Source Code:

```

void bubbleSortHariDesc(Tiket arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        for (int j = 0; j < n-i-1; j++) {
            if (arr[j].hari < arr[j+1].hari) {
                // tukar elemen
                Tiket temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

```

K. Fitur Selection Sort

Fitur ini berfungsi untuk memudahkan pengurutan Harga berdasarkan Ascending, dengan menggunakan algoritma Selection sort untuk analisis harga termurah hingga termahal.

Source Code:

```
void selectionSortHargaAsc(Tiket arr[], int panjang) {
    for (int i = 0; i < panjang - 1; i++) {
        int min = i;
        for (int j = i + 1; j < panjang; j++) {
            if (arr[j].totalHarga < arr[min].totalHarga) {
                min = j;
            }
        }
        // tukar elemen
        Tiket temp = arr[i];
        arr[i] = arr[min];
        arr[min] = temp;
    }
}
```

L. Fitur Insertion Sort

Fitur ini berfungsi untuk memudahkan pengurutan Durasi berdasarkan Ascending, dengan menggunakan algoritma Insertion sort untuk memudahkan melihat urutan durasi yang terpendek hingga terpanjang.

Source Code:

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

        // tukar elemen
        while (j >= 0 && arr[j].durasi > key.durasi) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

4. Uji Coba dan Hasil Output

4.1 Uji Coba dan Hasil Output

```
=====
| SELAMAT DATANG DI APLIKASI PEMBELIAN TIKET MINI PLAYGROUND |
=====
Silakan registrasi atau login untuk mulai menggunakan.
=====
|      MENU UTAMA      |
=====
1. Registrasi
2. Login
3. Keluar
Pilih Menu (1/2/3): 1
=== Registrasi ===
Masukkan Username: Nadia Rahmah
Masukkan Password: 2409106018
Registrasi berhasil! Silakan lanjut login.
```

Gambar 4.1 Registrasi

```
Registrasi berhasil! Silakan lanjut login.
=====
|      MENU UTAMA      |
=====
1. Registrasi
2. Login
3. Keluar
Pilih Menu (1/2/3): 2
=== Login ===
Masukkan Username: GAGAL
Masukkan Password: GAGAL
Username atau password salah! Coba lagi
=== Login ===
Masukkan Username: GAGAL
Masukkan Password: GAGAL
Username atau password salah! Coba lagi
=== Login ===
Masukkan Username: GAGAL
Masukkan Password: GAGAL
Anda sudah mencoba 3 Kali. Program berhenti.
PS C:\pratikum-apl\post-test\post-test-6>
```

Gambar 4.2 Login gagal

```

=====
|      MENU UTAMA      |
=====
1. Registrasi
2. Login
2. Login
3. Keluar
Pilih Menu (1/2/3): 2
Masukkan Username: Nadia Rahmah
Masukkan Password: 2409106018
Login berhasil! Selamat datang Nadia Rahmah!

```

Gambar 4.3 Login Berhasil

```

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 1
=== MENU TAMBAH TIKET ===
1. Weekend (+ 5.000)
2. Weekday
Masukkan Hari (1/2): 1
1. 1 Jam 20.000
2. 2 Jam 35.000
3. Sepuasnya 45.000
Masukkan Durasi (1/2/3): 1
***Jika tidak membawa kaus kaki, maka diwajibkan membeli seharga 10.000 ***
Apakah Anda membawa kaus kaki? (Y untuk Ya, T untuk Tidak): Y
Tiket berhasil ditambahkan!

```

Gambar 4.4 pilih Menu tiket 1

```

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 2
=== MENU TAMPILKAN TIKET ===

```

No	Hari	Durasi	Harga	Kaus Kaki	Total Harga	Status
1	Weekend	1 Jam	20000	Ya	25000	Aktif

Gambar 4.5 pilih Menu tiket 2


```
=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 3
=== MENU UBAH TIKET ===
=== MENU TAMPILKAN TIKET ===
      No      Hari      Durasi      Harga      Kaus Kaki      Total Harga      Status
      1      Weekend      1 Jam      20000      Ya      25000      Aktif
Masukkan nomor tiket yang ingin diubah: 1
1. Weekend
2. Weekday
Masukkan Hari baru (1/2): 2
1. 1 Jam 20.000
2. 2 Jam 35.000
3. Sepuasnya 45.000
Masukkan Durasi baru (1/2/3): 2
Apakah Anda membawa kaus kaki? (Y untuk Ya, T untuk Tidak): T
Masukkan status baru tiket: Non-Aktif
Tiket berhasil diubah!
=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 2
=== MENU TAMPILKAN TIKET ===
      No      Hari      Durasi      Harga      Kaus Kaki      Total Harga      Status
      1      Weekday      2 Jam      35000      Tidak      45000      Non-Aktif
```

Gambar 4.6 pilih menu tiket 3 dan 2

```
=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 4
=== MENU HAPUS TIKET ===
=== MENU TAMPILKAN TIKET ===
      No      Hari      Durasi      Harga      Kaus Kaki      Total Harga      Status
      1      Weekday      2 Jam      35000      Tidak      45000      Non-Aktif
      2      Weekday      Sepuasnya      45000      Ya      45000      Aktif
Masukkan nomor tiket yang ingin dihapus: 1
Tiket berhasil dihapus!
=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 2
=== MENU TAMPILKAN TIKET ===
      No      Hari      Durasi      Harga      Kaus Kaki      Total Harga      Status
      1      Weekday      Sepuasnya      45000      Ya      45000      Aktif
```

Gambar 4.7 pilih menu tiket 4 dan 2

```

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 0
Pilihan tidak valid

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 9
Pilihan tidak valid

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): -12
Pilihan tidak valid

=====
|      MENU TIKET      |
=====

```

Gambar 4.8 pilih menu tiket selain 1-6

```

=====
|      MENU TIKET      |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 5

=====
|      MENU SORTING     |
=====
1. Sorting berdasarkan Hari (Descending) - Bubble Sort
2. Sorting Harga (Ascending) - Selection Sort
3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort
4. Kembali ke Menu Tiket
Pilih metode sorting (1-4): █

```

Gambar 4.9 pilih menu tiket 5

Pilih menu (1/2/3/4/5): 2

=== MENU TAMPILKAN TIKET ===

No	Hari	Durasi	Harga	Kaus Kaki	Total Harga	Status
1	Weekday	Sepuasnya	45000	Ya	45000	Aktif
2	Weekend	2 Jam	35000	Tidak	50000	Aktif
3	Weekend	Sepuasnya	45000	Tidak	60000	Aktif
4	Weekday	Sepuasnya	45000	Ya	45000	Aktif
5	Weekday	1 Jam	20000	Tidak	30000	Aktif
6	Weekend	2 Jam	35000	Tidak	50000	Aktif
7	Weekday	1 Jam	20000	Ya	20000	Aktif

Gambar 4.10 Data tiket sebelum sorting

```
=====
|   MENU SORTING   |
=====
1. Sorting berdasarkan Hari (Descending) - Bubble Sort
2. Sorting Harga (Ascending) - Selection Sort
3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort
4. Kembali ke Menu Tiket
Pilih metode sorting (1-4): 1

Hasil Sorting Berdasarkan Hari (Weekend -> Weekday)
=== MENU TAMPILKAN TIKET ===
```

No	Hari	Durasi	Harga	Kaus Kaki	Total Harga	Status
1	Weekend	2 Jam	35000	Tidak	50000	Aktif
2	Weekend	Sepuasnya	45000	Tidak	60000	Aktif
3	Weekend	2 Jam	35000	Tidak	50000	Aktif
4	Weekday	Sepuasnya	45000	Ya	45000	Aktif
5	Weekday	Sepuasnya	45000	Ya	45000	Aktif
6	Weekday	1 Jam	20000	Tidak	30000	Aktif
7	Weekday	1 Jam	20000	Ya	20000	Aktif

Gambar 4.11 pilih menu sorting 1

```
=====
|   MENU SORTING   |
=====
1. Sorting berdasarkan Hari (Descending) - Bubble Sort
2. Sorting Harga (Ascending) - Selection Sort
3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort
4. Kembali ke Menu Tiket
Pilih metode sorting (1-4): 2

Hasil Sorting Berdasarkan Harga (Termurah -> Termahal)
=== MENU TAMPILKAN TIKET ===
```

No	Hari	Durasi	Harga	Kaus Kaki	Total Harga	Status
1	Weekday	1 Jam	20000	Ya	20000	Aktif
2	Weekday	1 Jam	20000	Tidak	30000	Aktif
3	Weekday	Sepuasnya	45000	Ya	45000	Aktif
4	Weekday	Sepuasnya	45000	Ya	45000	Aktif
5	Weekend	2 Jam	35000	Tidak	50000	Aktif
6	Weekend	2 Jam	35000	Tidak	50000	Aktif
7	Weekend	Sepuasnya	45000	Tidak	60000	Aktif

Gambar 4.12 pilih menu sorting 2

```
=====
|      MENU SORTING      |
=====
1. Sorting berdasarkan Hari (Descending) - Bubble Sort
2. Sorting Harga (Ascending) - Selection Sort
3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort
4. Kembali ke Menu Tiket
Pilih metode sorting (1-4): 3

Hasil Sorting Berdasarkan Durasi (Pendek -> Panjang)
=== MENU TAMPILKAN TIKET ===

```

No	Hari	Durasi	Harga	Kaus Kaki	Total Harga	Status
1	Weekday	1 Jam	20000	Ya	20000	Aktif
2	Weekday	1 Jam	20000	Tidak	30000	Aktif
3	Weekend	2 Jam	35000	Tidak	50000	Aktif
4	Weekend	2 Jam	35000	Tidak	50000	Aktif
5	Weekday	Sepuasnya	45000	Ya	45000	Aktif
6	Weekday	Sepuasnya	45000	Ya	45000	Aktif
7	Weekend	Sepuasnya	45000	Tidak	60000	Aktif

Gambar 4.13 pilih menu sorting 3

```
=====
|      MENU SORTING      |
=====
1. Sorting berdasarkan Hari (Descending) - Bubble Sort
2. Sorting Harga (Ascending) - Selection Sort
3. Sorting berdasarkan Durasi (Ascending) - Insertion Sort
4. Keluar ke Menu Tiket
Pilih metode sorting (1-4): 4

=====
|      MENU TIKET        |
=====
1. Tambah Tiket
```

Gambar 4.14 pilih menu sorting 4

```
=====
|      MENU TIKET        |
=====
1. Tambah Tiket
2. Tampilkan Tiket
3. Ubah data Tiket
4. Hapus Tiket
5. Menu Sorting tiket
6. Keluar ke menu utama
Pilih menu (1/2/3/4/5): 6

=====
|      MENU UTAMA        |
=====
1. Registrasi
```

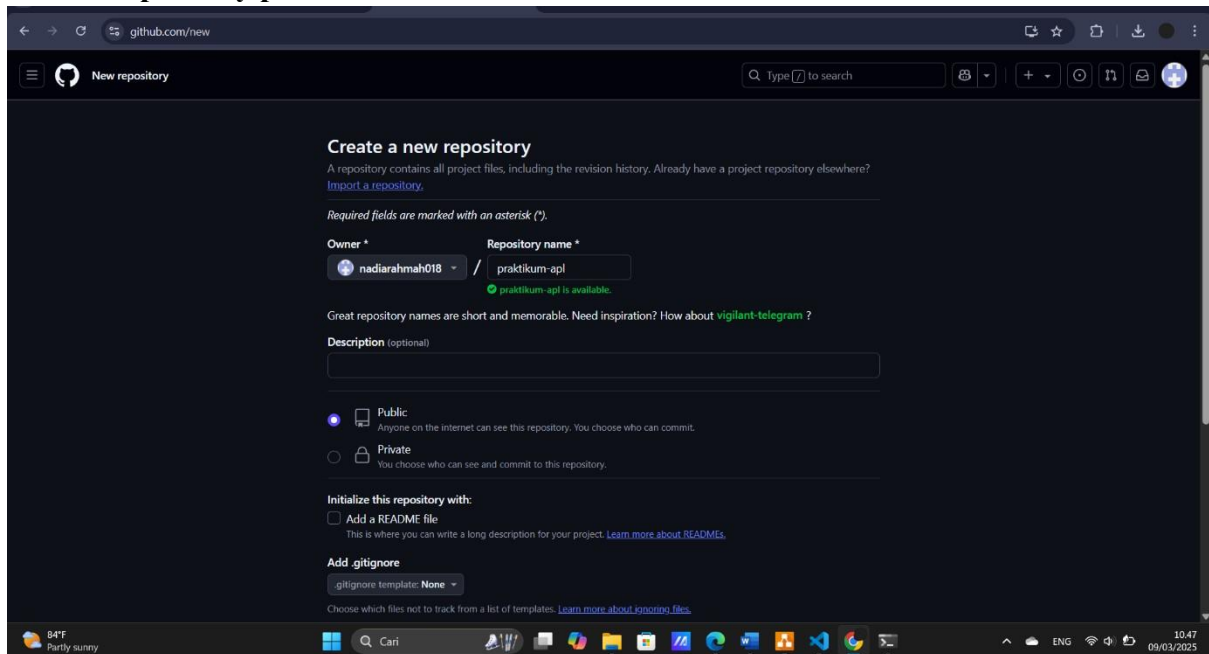
Gambar 4.15 pilih menu tiket 6

```
=====
|      MENU UTAMA      |
=====
1. Registrasi
2. Login
3. Keluar
Pilih Menu (1/2/3): 3
Terima Kasih!
PS C:\pratikum-apl\post-test\post-test-6>
```

Gambar 4.16 pilih menu utama 3

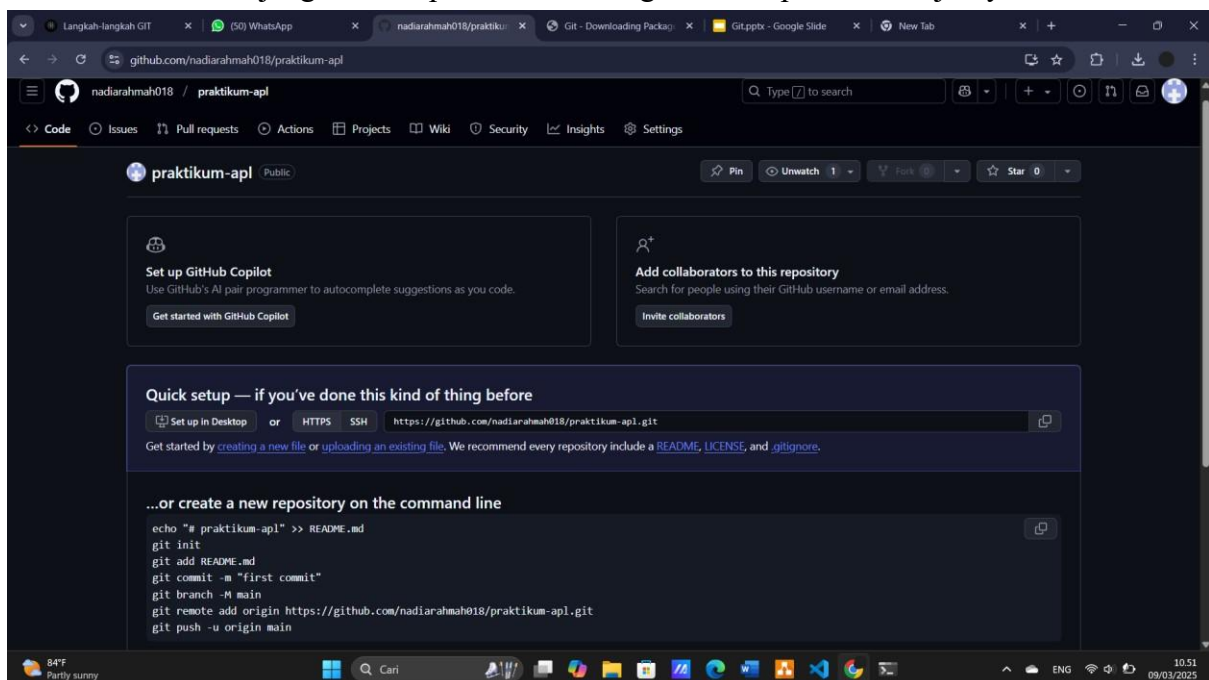
5. Langkah-Langkah GIT

1. Buat Repository pada Github



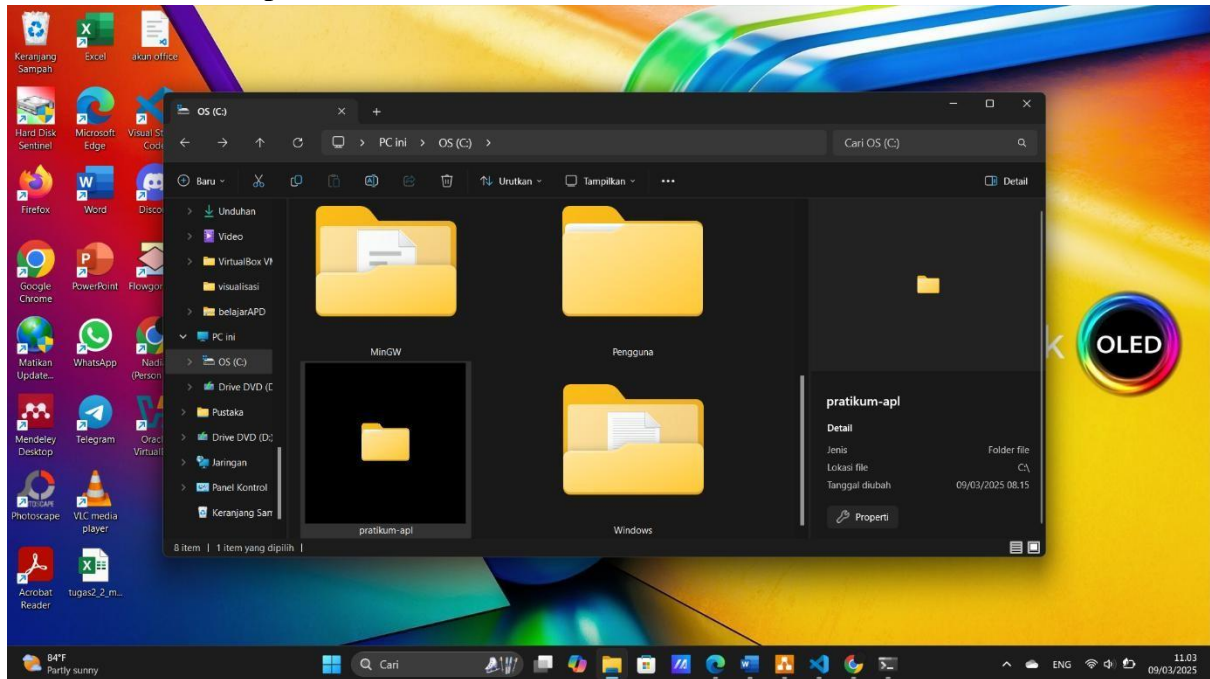
Gambar 5.1 Membuat Repository

Isi Nama repository yaitu praktikum-apl. Tetapkan Nama repository bersifat public. Kemudian penting memperhatikan Langkah-langkah command yang akan dilakukan di terminal. Tab Github di bawah ini jangan ditutup karena akan digunakan diproses selanjutnya.



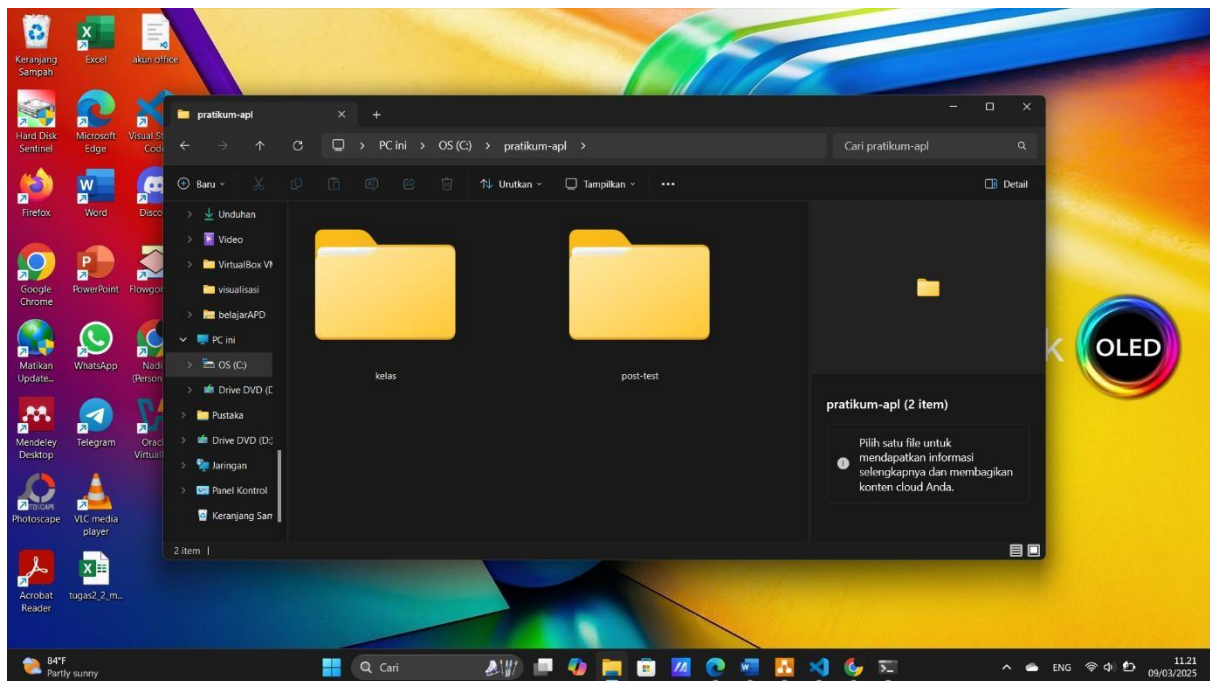
Gambar 5.2 Repository APL

2. Buat folder di Explorer



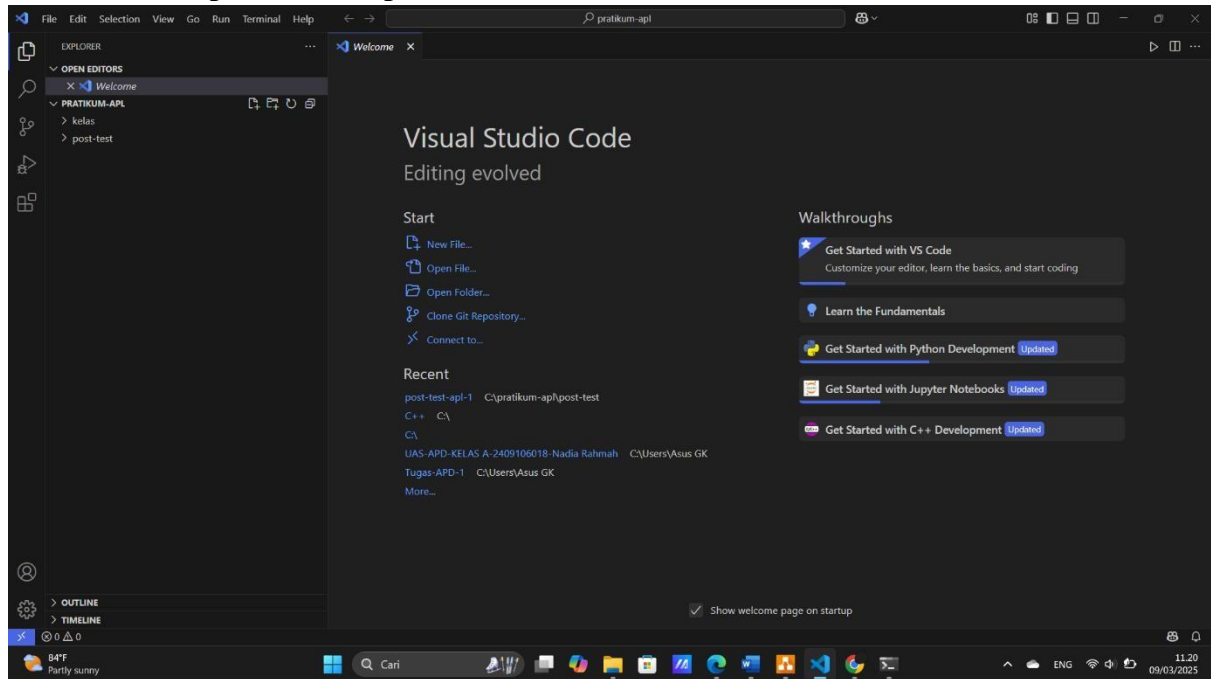
Gambar 5.3 Folder praktikum-apl

Buat folder pada explorer masing-masing dengan nama praktikum-apl sesuai nama repository Github. Selanjutnya tambahkan 2 folder lagi yang diberi nama kelas dan post-test di dalam folder praktikum-apl.



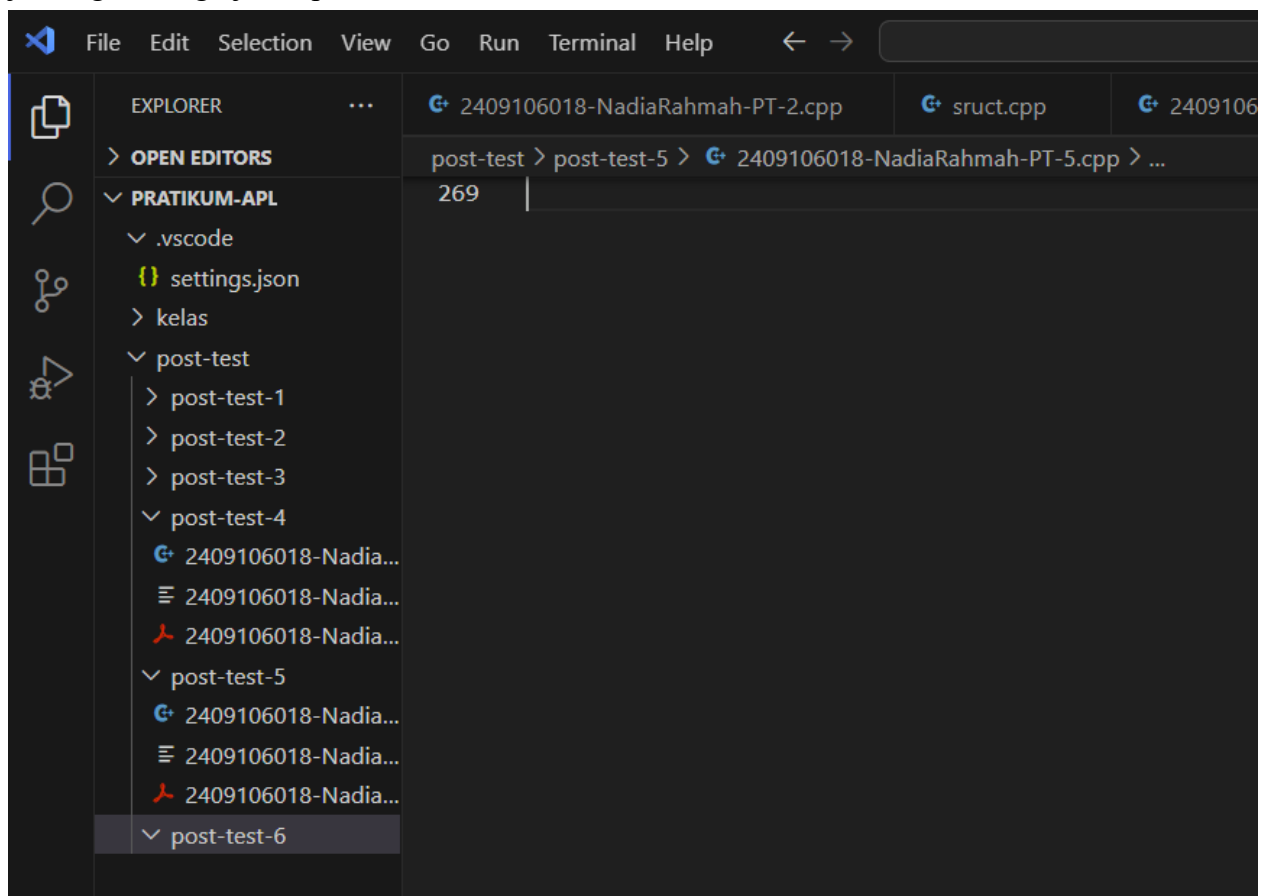
Gambar 5.4 Folder kelas dan post-test

3. Buka Folder praktikum-apl di VSCode

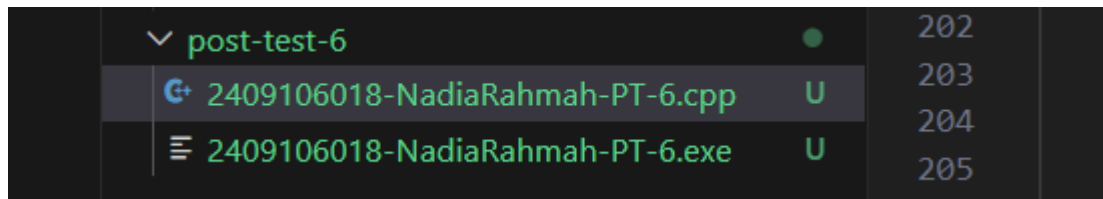


Gambar 5.5 VSCode

4. Buat Folder post-test-x di Folder post-test Contoh jika ingin mengerjakan post-test-6.



Gambar 5.6 Folder post-test-6

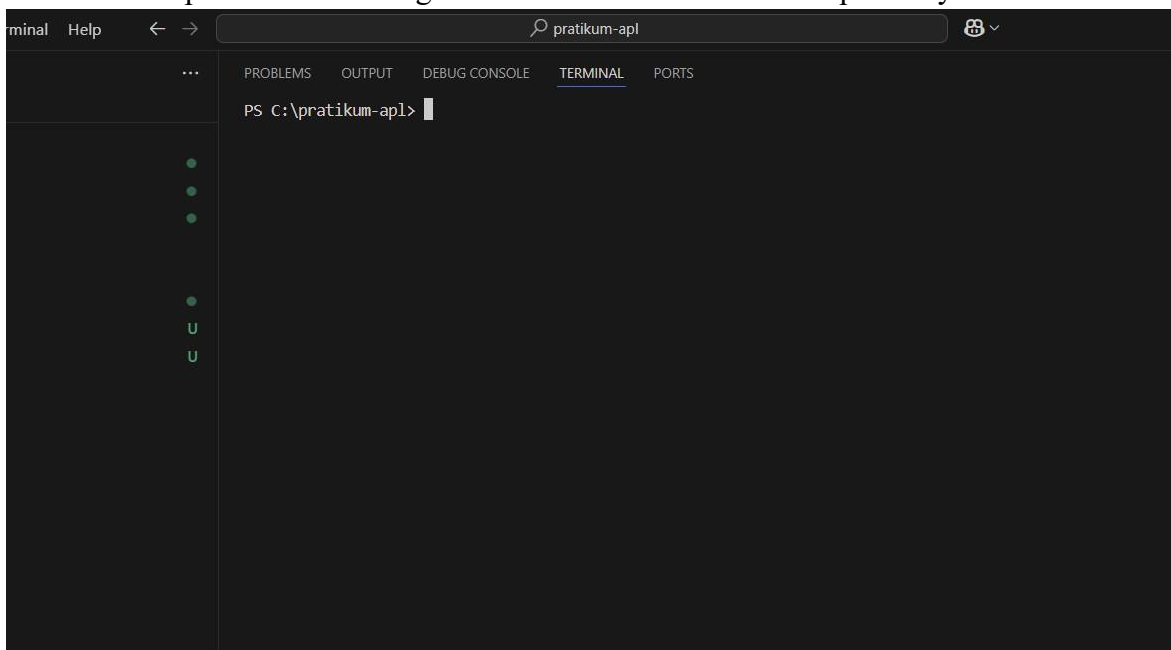


Gambar 5.7 File post-test-6

Kemudian pada folder post-test-5, buat file yang diperlukan dengan nama yang ditentukan.

5. Buka Terminal

Buka terminal pada VSCode dengan cara menekan tombol `ctrl+~` pada keyboard

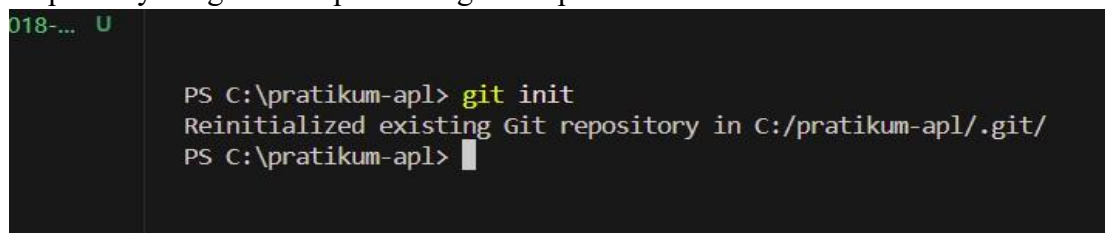


Gambar 5.8 Terminal VSCode

Pastikan path sesuai, jika path belum sesuai, naik folder paling atas dengan cara ketik (`cd ../../` atau `cd ..`).

6. Git Init

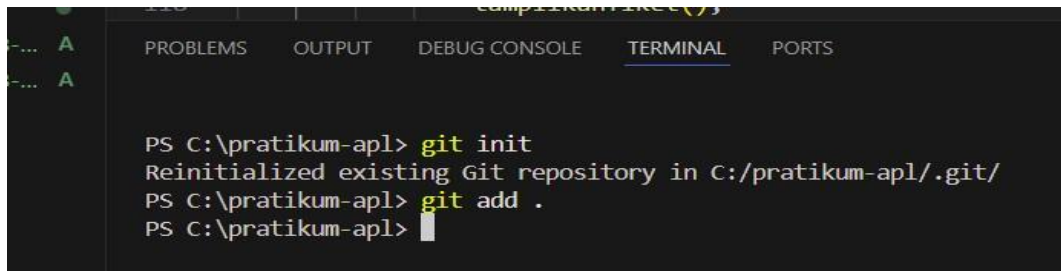
Inisiasi repository dengan ketik perintah 'git init' pada terminal.



Gambar 5.9 Git Init

7. Git Add

Lakukan git add untuk menambahkan file apa saja yang ingin kita commit. Kemudian. Ketikkan Perintah 'git add atau .' pada terminal.

A screenshot of a terminal window with a dark background. The terminal shows the following commands and output:

```
PS C:\pratikum-apl> git init
Reinitialized existing Git repository in C:/pratikum-apl/.git/
PS C:\pratikum-apl> git add .
PS C:\pratikum-apl>
```

 The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), and PORTS.

Gambar 5.10 Git Add

8. Git Commit

Melakukan git commit untuk membuat cekpoint. Ketik perintah git commit -m "Pesan commit" pada terminal.

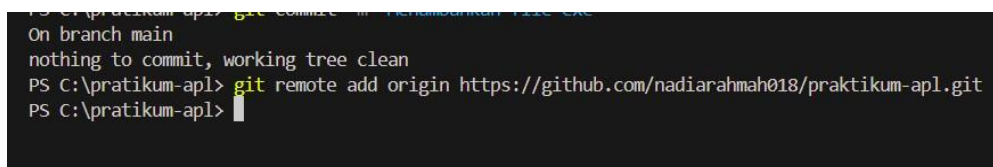
A screenshot of a terminal window showing the following command and output:

```
PS C:\pratikum-apl\post-test> git commit -m "Menambahkan file cpp dan exe"
[main 2d48873] Menambahkan file cpp dan exe
2 files changed, 3 insertions(+), 3 deletions(-)
```

Gambar 5.11 Git Commit

9. Git Remote

Melakukan git remote untuk menghubungkan repository yang ada di local komputer dengan repository cloud pada Github. Copy git remote yang ada pada github, kemudian paste pada terminal.

A screenshot of a terminal window showing the following commands and output:

```
PS C:\pratikum-apl> git commit -m "Menambahkan file exe"
On branch main
nothing to commit, working tree clean
PS C:\pratikum-apl> git remote add origin https://github.com/nadiarahmah018/praktikum-apl.git
PS C:\pratikum-apl>
```

Gambar 5.11 Git Remote

10. Git Push

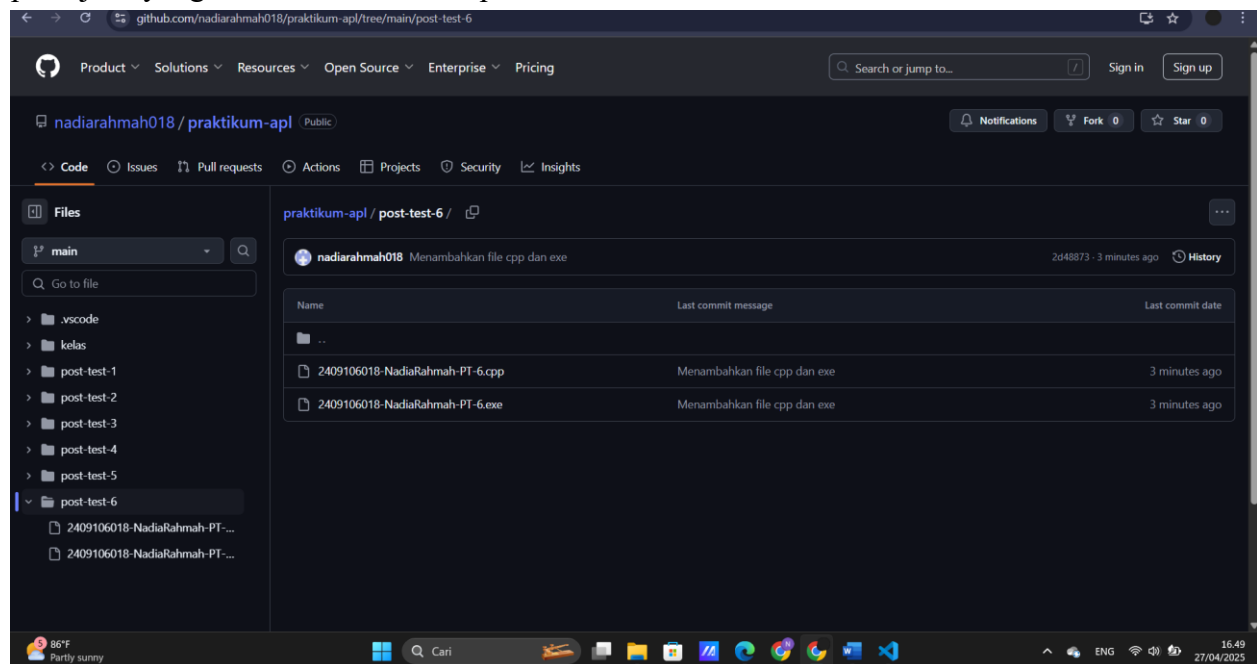
Git push adalah perintah yang digunakan untuk mengirimkan (meng-upload) perubahan yang telah di commit di repository lokal ke repository jarak jauh(remote repository). Ketik git push -u origin main pada terminal.

```
PS C:\pratikum-apl\post-test> git push -u origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 789 bytes | 263.00 KiB/s, done.
Total 7 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To https://github.com/nadiarahmah018/praktikum-apl.git
    93b54c4..2d48873  main -> main
branch 'main' set up to track 'origin/main'.
PS C:\pratikum-apl\post-test> |
```

Gambar 5.12 Git Push

11. Reload Tab Github di Browser

Setelah semua langkah-langkah selesai, reload tab Github di browser. Maka tampilan semua pekerjaan yang dilakukan akan tampil di Github.



Gambar 5.13 File Github