

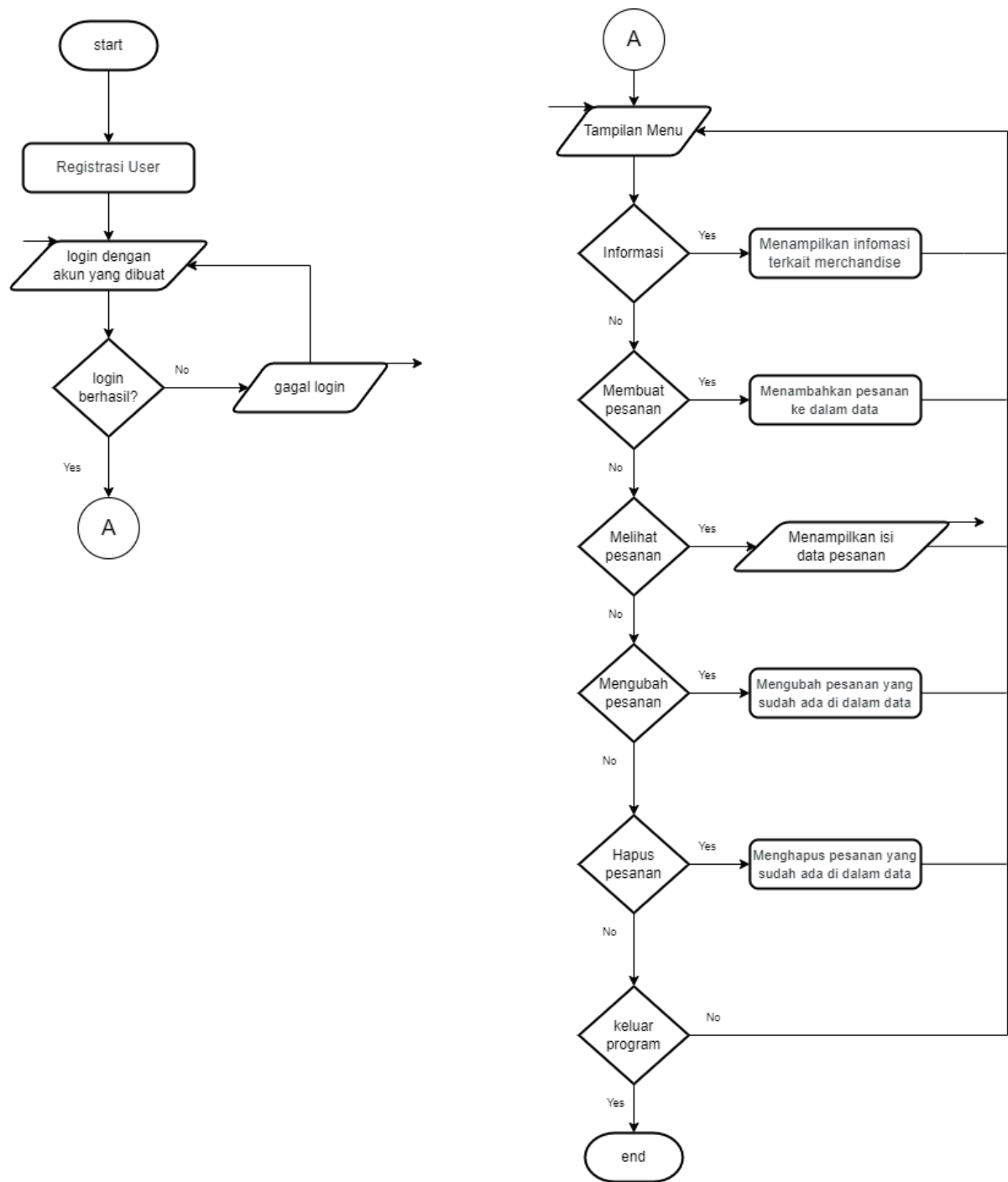
LAPORAN PRAKTIKUM
POSTTEST 4
ALGORITMA PEMROGRAMAN LANJUT



Disusun oleh:
Ken Bilqis Nuraini (2409106015)
Kelas A1 '24

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

1. Flowchart



Gambar 1.1 Flowchart

2. Analisis Program

Program ini merupakan program pemesanan merchandise band The Jansen. di program ini user dapat melihat informasi terkait merchandise, memesan merchandise yang ingin dibeli, melihat nomor pemesanan, mengubah pesanan dan menghapus pesanan. Sebelum masuk pada menu utama user diminta untuk membuat akun dengan menggunakan username dan password. Setelah itu user melakukan login dengan menginput username dan password yang sudah dibuat sebelumnya, Jika gagal sebanyak tiga kali maka program akan berhenti otomatis. Setelah berhasil login, user dapat menggunakan semua fitur yang ada.

3. Source Code

A. Prosedur Register

Fitur ini digunakan untuk membuat akun untuk login

Source Code:

```
void prosesRegistrasi(Akun daftarAkun[], int &totalAkun) {
    if (totalAkun < maksimalPesanan) {
        cout << "\nMasukkan Username: ";
        getline(cin, daftarAkun[totalAkun].username);
        cout << "Masukkan Password: ";
        getline(cin, daftarAkun[totalAkun].password);
        totalAkun++;
        cout << "\n===== " << endl;
        cout << "== Akun berhasil didaftarkan ==" << endl;
        cout << "===== " << endl;
    } else {
        cout << "\n===== " << endl;
        cout << "== Kuota akun sudah penuh! ==" << endl;
        cout << "===== " << endl;
    }
}
```

Gambar 3.1 Fitur Register

B. Fungsi Login

Fitur ini digunakan untuk login dengan menginput data yang sudah dibuat pada fitur register. Jika input benar, akan menampilkan pesan login berhasil dan lanjut ke menu utama program. Sedangkan jika tidak sesuai, user diberikan kesempatan hingga 3 kali untuk mencoba kembali sebelum program berhenti.

Source Code:

```
bool prosesLogin(Akun daftarAkun[], int totalAkun, int &indeksPenggunaAktif, int
&kesempatanLogin) {
    string username, password;
    cout << "\nMasukkan Username: ";
    getline(cin, username);
    cout << "Masukkan Password: ";
    getline(cin, password);

    for (int i = 0; i < totalAkun; i++) {
```

```

        if (daftarAkun[i].username == username && daftarAkun[i].password ==
password) {
            indeksPenggunaAktif = i;
            cout << "\n===== " << endl;
            cout << "= Login berhasil :D =" << endl;
            cout << "===== " << endl;
            return true;
        }
    }

    cout << "\n===== "
<< endl;
    cout << "== !!!Username atau Password yang dimasukkan salah!!! ==" <<
endl;
    cout << "===== " <<
endl;
    kesempatanLogin++;

    if (kesempatanLogin == 3) {
        cout << "\n===== " << endl;
        cout << "== Maaf kesempatan login anda habis :( ==" << endl;
        cout << "===== " << endl;
        exit(0);
    }

    return false;
}

```

Gambar 3.2 Fitur Login

C. Prosedur Menampilkan Informasi Merchandise

Fitur ini berfungsi untuk menampilkan data informasi merchandise pada array 2 dimensi

Source Code:

```

void tampilkanInformasiMerchandise() {
    string infoMerchandise[3][2] = {
        {"T-Shirt", "Rp 150.000"},
        {"Hoodie", "Rp 300.000"},
        {"Topi", "Rp 80.000"}
    };

    cout << "\n== Informasi Merchandise ==" << endl;
    cout << "\n" << setw(10) << "Item" << setw(15) << "Harga\n";
    for (int i = 0; i < 3; i++) {
        cout << setw(10) << infoMerchandise[i][0] << setw(15) <<
infoMerchandise[i][1] << "\n";
    }
}

```

```

    cout << "\n== Ukuran Tersedia: S/M/L/XL ==\n";
}

```

Gambar 3.3 Fitur Menampilkan Informasi Merchandise

D. Prosedur Membuat Pesanan

Fitur ini berfungsi untuk user membuat pesanan dengan mengisi data yang diminta lalu program memasukkannya ke dalam array

Source Code:

```

void buatPesananBaru(Akun &akunPengguna) {
    if (akunPengguna.itemPesanan < maksimalPesanan) {
        cout << "\n== Buat Pesanan ==\n" << endl;
        cout << "\nMasukkan Nama Pemesan: ";
        getline(cin,
akunPengguna.daftarPesanan[akunPengguna.itemPesanan].namaPemesan);
        cout << "Masukkan Nama Item: ";
        cin >> akunPengguna.daftarPesanan[akunPengguna.itemPesanan].namaItem;
        cin.ignore();
        cout << "Masukkan Ukuran [S/M/L/XL]: ";
        getline(cin,
akunPengguna.daftarPesanan[akunPengguna.itemPesanan].ukuranPesanan);
        akunPengguna.itemPesanan++;
        cout << "\n===== " <<
endl;
        cout << "== Terimakasih, Pesanan Berhasil ditambahkan :D ==\n" << endl;
        cout << "===== " << endl;
    } else {
        cout << "\n===== " << endl;
        cout << "== Mohon Maaf Kuota Pesanan Sudah Penuh :( ==\n" << endl;
        cout << "===== " << endl;
    }
}

```

Gambar 3.4 Fitur Membuat Pesanan

E. Prosedur Menampilkan Data Pesanan

Fitur ini berfungsi untuk menampilkan semua pesanan pada data array

Source Code:

```

void lihatDaftarPesanan(const Akun &akunPengguna) {
    cout << "\n===== Daftar Pesanan =====\n";
    if (akunPengguna.itemPesanan == 0) {
        cout << "\n===== " << endl;
        cout << "== Mohon Maaf Belum Ada Pesanan :( ==\n" << endl;
    }
}

```

```

        cout << "=====" << endl;
    } else {
        cout << setw(5) << "No" << setw(20) << "Nama Pemesan" << setw(15) <<
"item" << setw(10) << "Ukuran\n";
        for (int i = 0; i < akunPegguna.itemPesanan; i++) {
            tampilkanPesanan(akunPegguna.daftarPesanan[i], i+1); // Menggunakan
fungsi overloading
        }
    }
}

```

Gambar 3.5 Fitur Menampilkan Data Pesanan

F. Prosedur Mengubah Pesanan

Fitur ini berfungsi untuk mengubah data yang sudah ada dalam array

Source Code:

```

void ubahPesanan(Akun &akunPegguna) {
    lihatDaftarPesanan(akunPegguna);

    if (akunPegguna.itemPesanan == 0) return;

    cout << "\n=== Ubah Pesanan ===\n";
    cout << "\nMasukkan Nomor Pesanan yang ingin diubah: ";
    int nomorPesanan;
    cin >> nomorPesanan;
    cin.ignore();

    if (nomorPesanan > 0 && nomorPesanan <= akunPegguna.itemPesanan) {
        cout << "\nMasukkan Nama Pemesan Baru: ";
        getline(cin, akunPegguna.daftarPesanan[nomorPesanan - 1].namaPemesan);
        cout << "Masukkan Nama Item Baru: ";
        cin >> akunPegguna.daftarPesanan[nomorPesanan - 1].namaItem;
        cin.ignore();
        cout << "Masukkan Ukuran Baru (S/M/L/XL): ";
        getline(cin, akunPegguna.daftarPesanan[nomorPesanan -
1].ukuranPesanan);
        cout << "\n=====" << endl;
        cout << "== Pesanan berhasil diubah :D ==" << endl;
        cout << "=====" << endl;
    } else {
        cout << "\n=====" << endl;
        cout << "== Nomor pesanan tidak valid! ==" << endl;
        cout << "=====" << endl;
    }
}

```

Gambar 3.6 Fitur Mengubah Pesanan

G. Prosedur Menghapus Pesanan

Fitur ini berfungsi untuk menghapus data yang sudah ada dalam array

Source Code:

```
void hapusPesanan(Akun &akunPegguna) {
    lihatDaftarPesanan(akunPegguna);

    if (akunPegguna.itemPesanan == 0) return;

    cout << "\n=== Hapus Pesanan ===\n";
    cout << "\nMasukkan Nomor Pesanan yang ingin dihapus: ";
    int nomorPesanan;
    cin >> nomorPesanan;
    cin.ignore();

    if (nomorPesanan > 0 && nomorPesanan <= akunPegguna.itemPesanan) {
        for (int i = nomorPesanan - 1; i < akunPegguna.itemPesanan - 1; i++) {
            akunPegguna.daftarPesanan[i] = akunPegguna.daftarPesanan[i + 1];
        }
        akunPegguna.itemPesanan--;
        cout << "\n===== " << endl;
        cout << "== Pesanan berhasil dihapus :D == " << endl;
        cout << "===== " << endl;
    } else {
        cout << "\n===== " << endl;
        cout << "== Nomor pesanan tidak valid! == " << endl;
        cout << "===== " << endl;
    }
}
```

Gambar 3.7 Fitur Menghapus Pesanan

H. Fungsi Rekursif

Fungsi ini berfungsi untuk menghitung total pesanan

Source Code:

```
int hitungTotalPesananRekursif(const Akun &akunPegguna, int indeks) {
    if (indeks >= akunPegguna.itemPesanan) {
        return 0;
    }
    return 1 + hitungTotalPesananRekursif(akunPegguna, indeks + 1);
}
```

Gambar 3.8 Fungsi Rekursif

H. Fungsi Overloading

Fungsi ini berfungsi untuk menampilkan pesanan

Source Code:

```
// Fungsi overloading untuk menampilkan pesanan (tanpa nomor)
void tampilkanPesanan(const Pesanan &pesanan) {
    cout << setw(20) << pesanan.namaPemesan << setw(15) << pesanan.namaItem <<
    setw(10) << pesanan.ukuranPesanan << "\n";
}

// Fungsi overloading untuk menampilkan pesanan (dengan nomor)
void tampilkanPesanan(const Pesanan &pesanan, int nomor) {
    cout << setw(5) << nomor << setw(20) << pesanan.namaPemesan << setw(15) <<
    pesanan.namaItem << setw(10) << pesanan.ukuranPesanan << "\n";
}
```

Gambar 3.9 Fungsi Rekursif

4. Uji Coba dan Hasil Output

4.1 Hasil Output

```
=====
==  SELAMAT DATANG DI PROGRAM PREORDER  ==
==    MERCHANDISE BAND THE JANSEN    ==
=====

Silahkan pilih menu:
1. Register
2. Login
3. Keluar
Masukkan pilihan[1-3]: 1

Masukkan Username: Ken
Masukkan Password: 015

=====
==  Akun berhasil didaftarkan  ==
=====
```

Gambar 4.1 Register

```
Silahkan pilih menu:
1. Register
2. Login
3. Keluar
Masukkan pilihan[1-3]: 2

Masukkan Username: Ken
Masukkan Password: 015

=====
=  Login berhasil :D  =
=====
```

Gambar 4.2 Login

```

=====
==      MENU UTAMA PROGRAM      ==
==  1. Informasi Merchandise  ==
==  2. Buat Pesanan           ==
==  3. Lihat Pesanan          ==
==  4. Ubah Pesanan           ==
==  5. Hapus Pesanan          ==
==  6. Logout                 ==
=====

Masukkan pilihan[1-6]: 1

==      Informasi Merchandise      ==

      Item          Harga
T-Shirt    Rp 150.000
Hoodie     Rp 300.000
Topi       Rp 80.000

== Ukuran Tersedia: S/M/L/XL ==

```

Gambar 4.3 Menampilkan Informasi Merchandise

```

=====
==      MENU UTAMA PROGRAM      ==
==  1. Informasi Merchandise  ==
==  2. Buat Pesanan           ==
==  3. Lihat Pesanan          ==
==  4. Ubah Pesanan           ==
==  5. Hapus Pesanan          ==
==  6. Logout                 ==
=====

Masukkan pilihan[1-6]: 2

==      Buat Pesanan      ==

Masukkan Nama Pemesan: Ken
Masukkan Nama Item: Topi
Masukkan Ukuran [S/M/L/XL]: S

=====
==      Terimakasih, Pesanan Berhasil ditambahkan :D      ==
=====

```

Gambar 4.4 Membuat pesanan

```

=====
==      MENU UTAMA PROGRAM      ==
==  1. Informasi Merchandise  ==
==  2. Buat Pesanan           ==
==  3. Lihat Pesanan          ==
==  4. Ubah Pesanan           ==
==  5. Hapus Pesanan          ==
==  6. Logout                 ==
=====

Masukkan pilihan[1-6]: 3

===== Daftar Pesanan =====
  No      Nama Pemesan      Jumlah  Ukuran
   1             Ken         Topi        S

```

Gambar 4.5 Menampilkan Data Pesanan

```

=====
==      MENU UTAMA PROGRAM      ==
==  1. Informasi Merchandise  ==
==  2. Buat Pesanan           ==
==  3. Lihat Pesanan          ==
==  4. Ubah Pesanan           ==
==  5. Hapus Pesanan          ==
==  6. Logout                 ==
=====

Masukkan pilihan[1-6]: 4

===== Daftar Pesanan =====
  No      Nama Pemesan      Jumlah  Ukuran
   1             Ken         Topi        S

=== Ubah Pesanan ===

Masukkan Nomor Pesanan yang ingin diubah: 1

Masukkan Nama Pemesan Baru: Bil
Masukkan Nama Item Baru: Hoodie
Masukkan Ukuran Baru (S/M/L/XL): XL

=====
==  Pesanan berhasil diubah :D  ==
=====

```

Gambar 4.6 Mengubah Pesanan

```

=====
==  MENU UTAMA PROGRAM  ==
== 1. Informasi Merchandise ==
== 2. Buat Pesanan      ==
== 3. Lihat Pesanan     ==
== 4. Ubah Pesanan      ==
== 5. Hapus Pesanan     ==
== 6. Logout            ==
=====

Masukkan pilihan[1-6]: 5

===== Daftar Pesanan =====
   No      Nama Pemesan      Jumlah   Ukuran
   1        Bil             Hoodie    XL

=== Hapus Pesanan ===

Masukkan Nomor Pesanan yang ingin dihapus: 1

=====
== Pesanan berhasil dihapus :D ==
=====

```

Gambar 4.7 Menghapus Pesanan

```

=====
==  MENU UTAMA PROGRAM  ==
== 1. Informasi Merchandise ==
== 2. Buat Pesanan      ==
== 3. Lihat Pesanan     ==
== 4. Ubah Pesanan      ==
== 5. Hapus Pesanan     ==
== 6. Logout            ==
=====

Masukkan pilihan[1-6]: 6

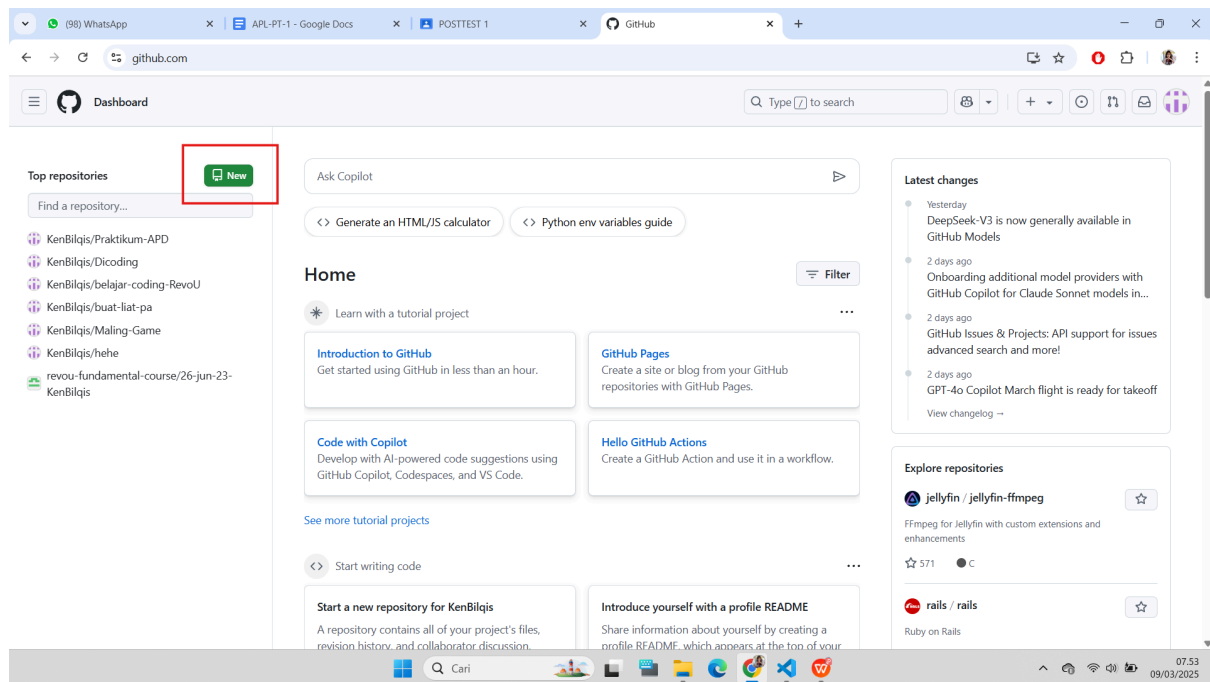
=====
== Terimakasih Telah Menggunakan Program ini ==
== Mohon maaf apabila terdapat kekurangan :) ==
=====

```

Gambar 4.8 Keluar dari Program

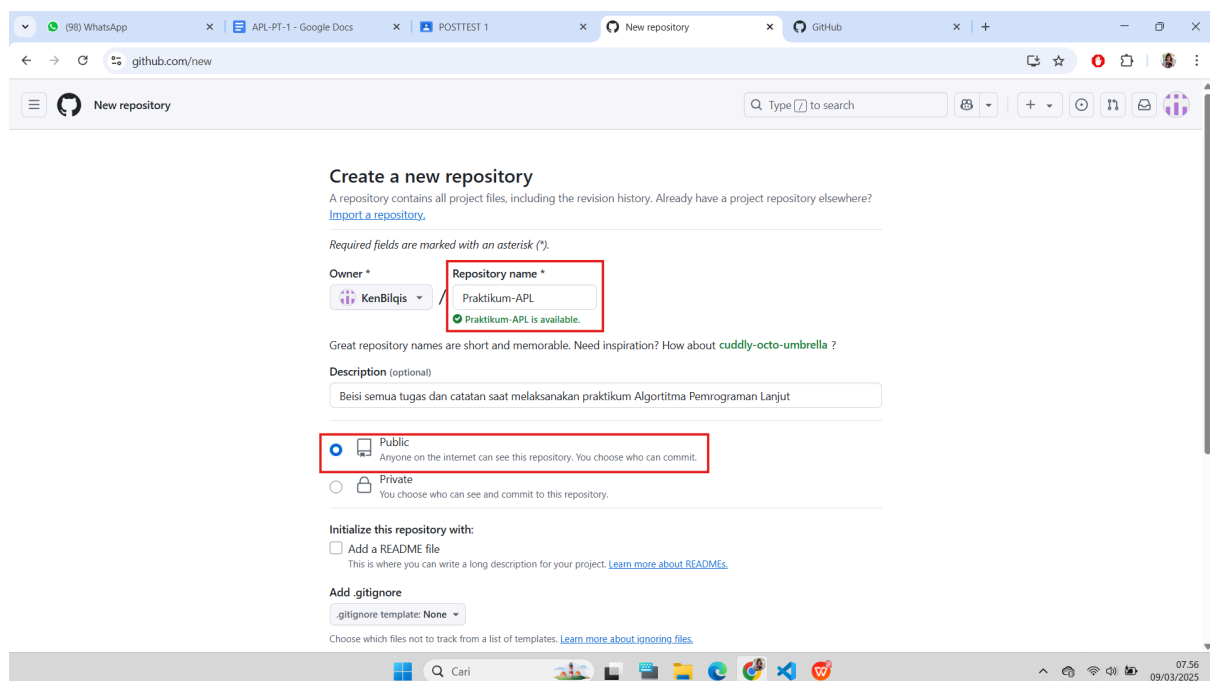
5. Langkah-langkah GIT

1. Membuat Repository pada Github



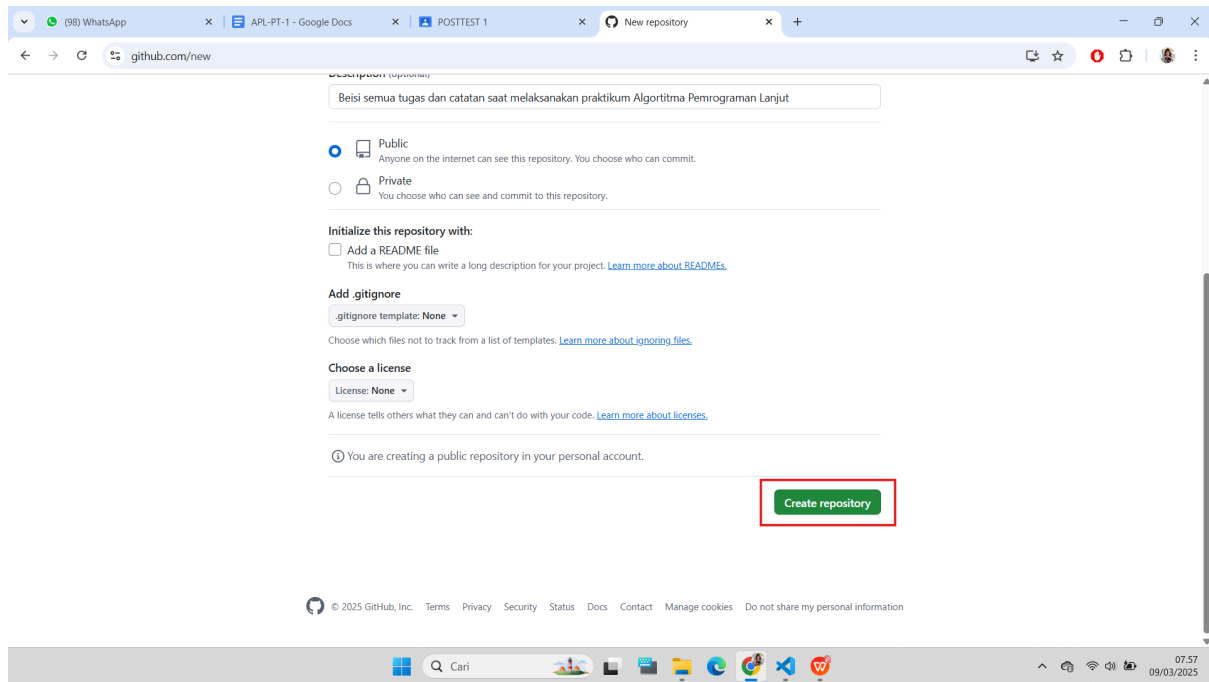
Gambar 5.1 Membuat Repository

Buat repository dengan mengklik 'New' pada halaman dashboard.



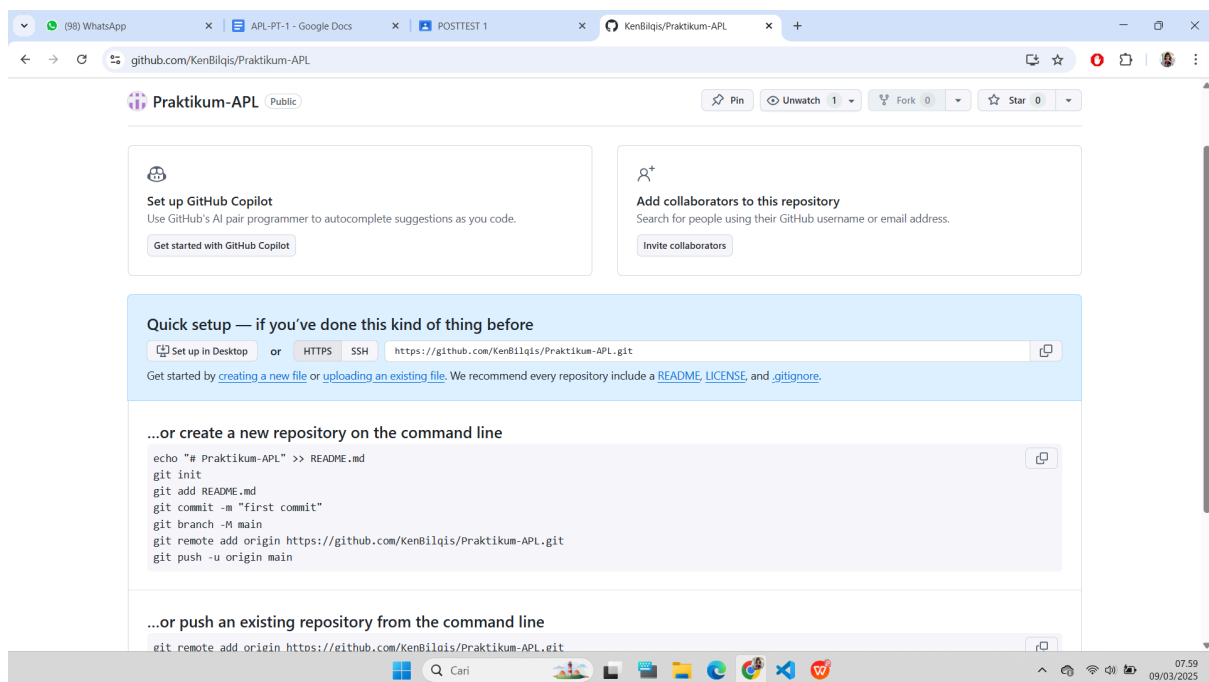
Gambar 5.2 Mengatur Repository

Masukkan nama repository sesuai dengan ketentuan posttest, dan atur repository menjadi publik. Untuk bagian deskripsi boleh diisi boleh tidak.



Gambar 5.3 Create Repository

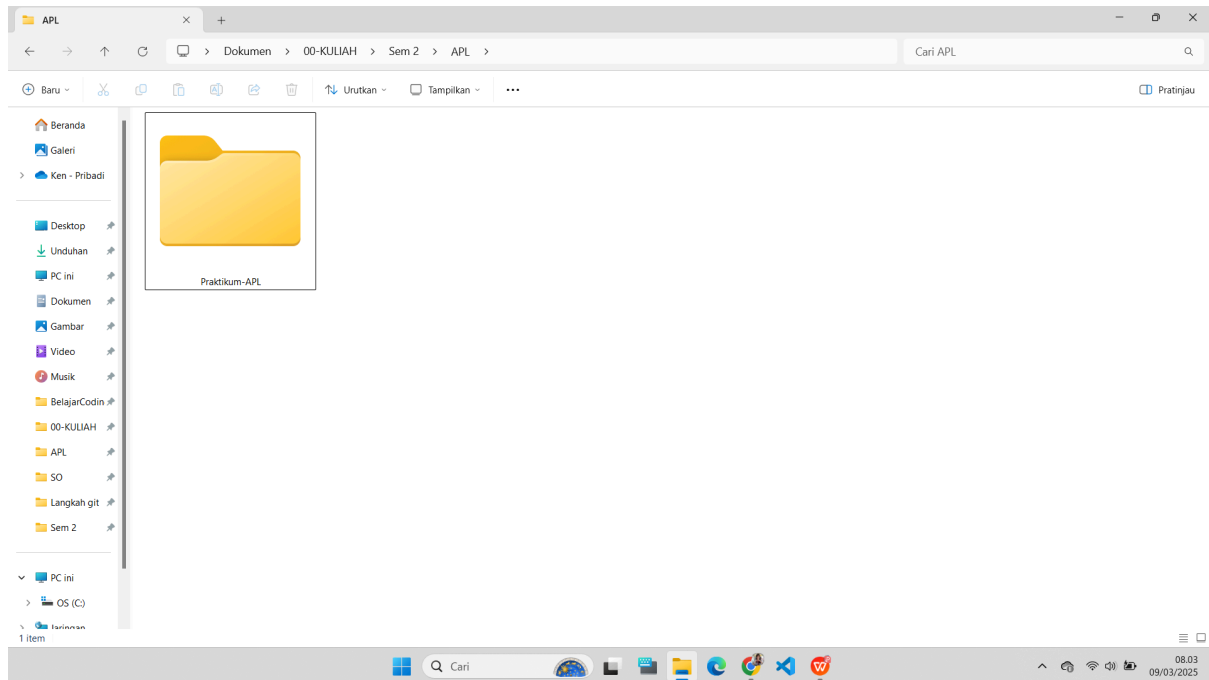
Klik create repository untuk membuat repository.



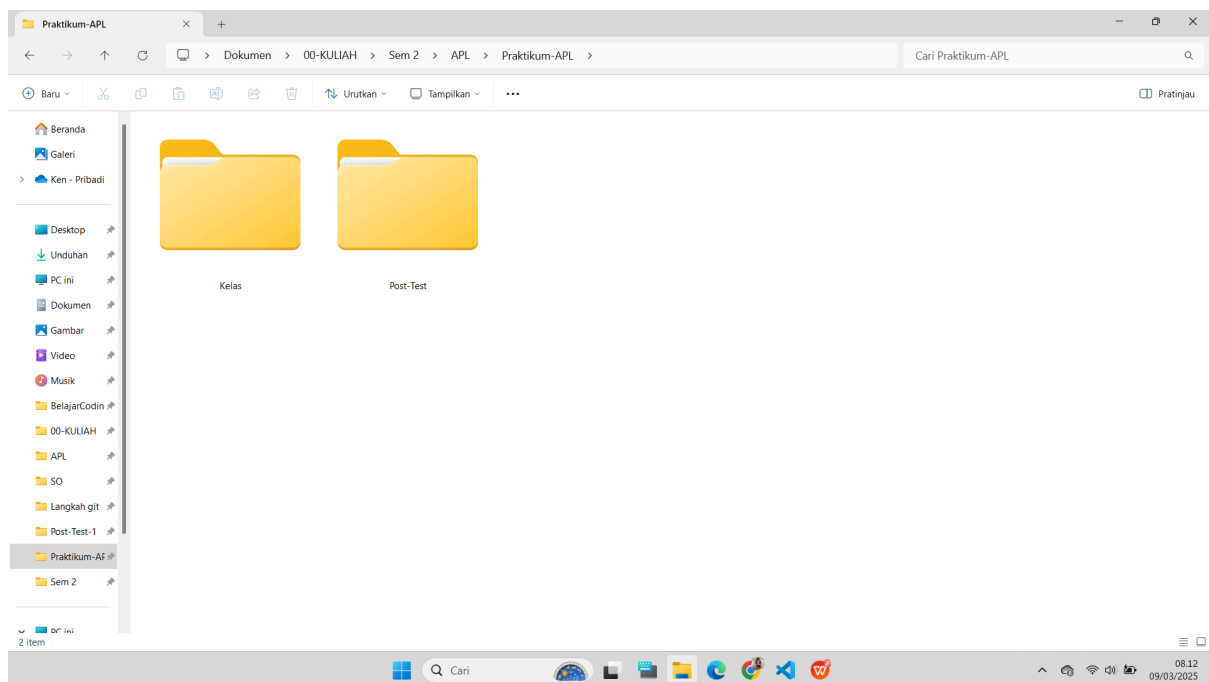
Gambar 5.4 Repository

Setelah itu untuk tab ini jangan ditutup karena kita masih akan menggunakannya sampai selesai.

2. Membuat Folder di Explorer



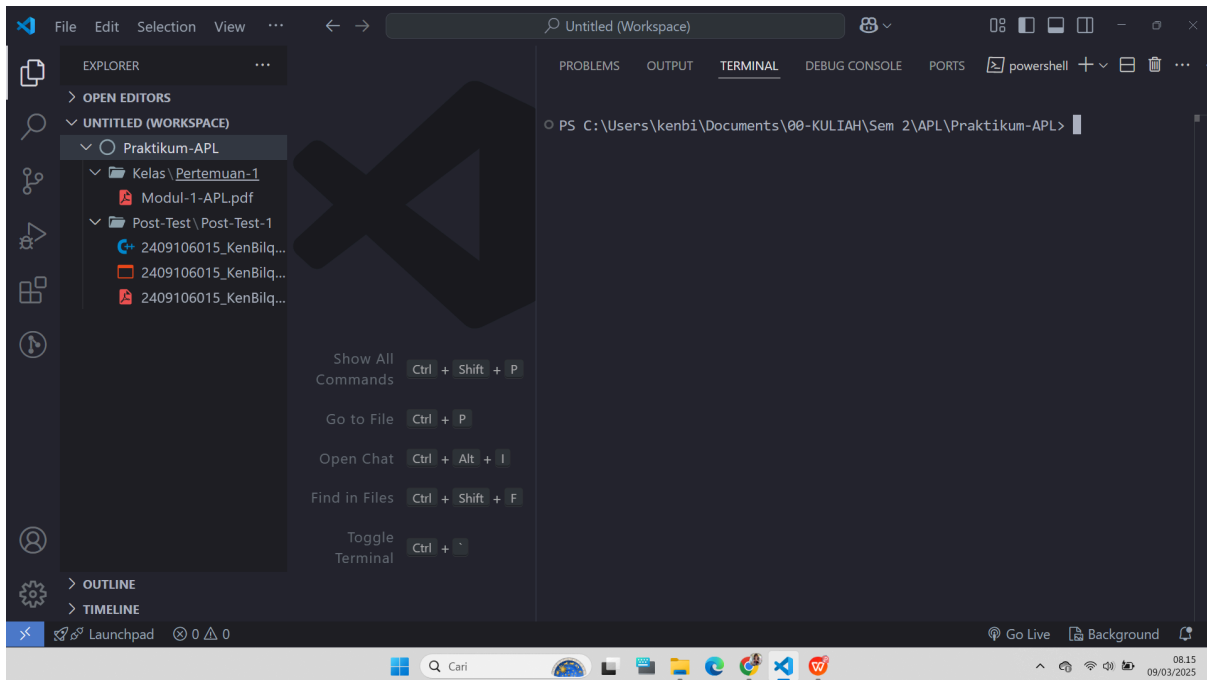
Gambar 5.5 Membuat Folder



Gambar 5.6 Isi Folder

Buat folder pada file explorer sesuai dengan ketentuan posttest.

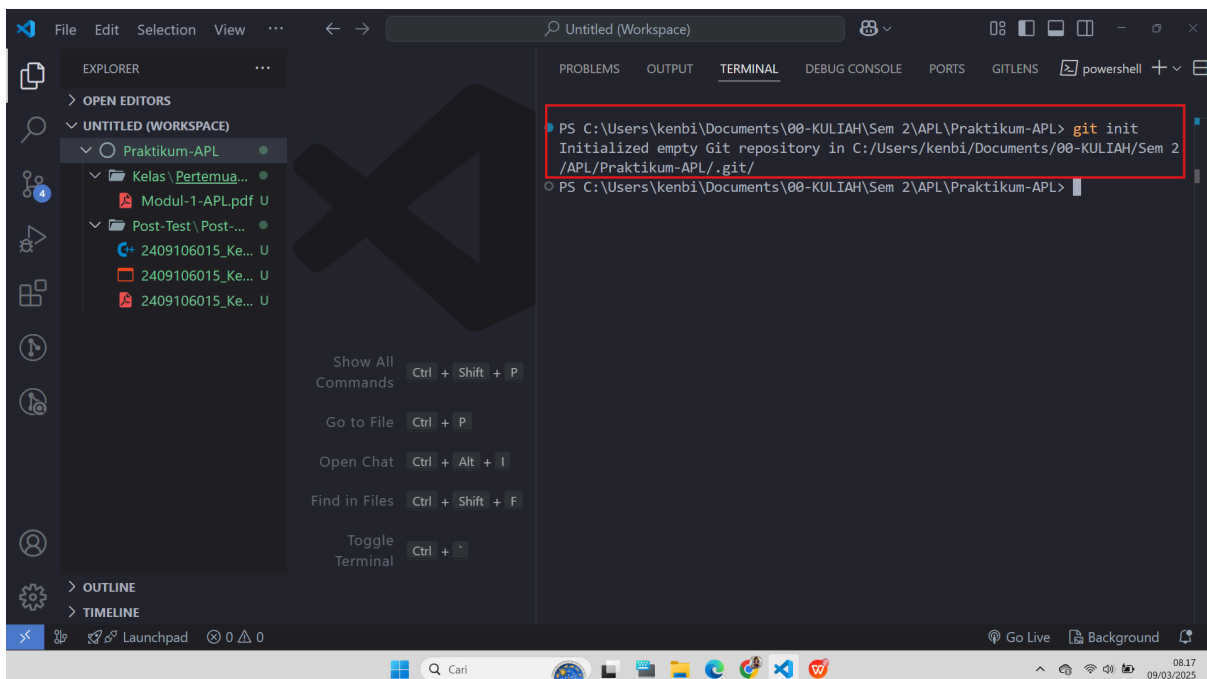
3. Buka Folder di VSCode



Gambar 5.7 Terminal VSCode

Buka folder yang sudah di buat di VSCode, lalu buka terminal dengan menekan tombol **Ctrl+`** pada keyboard. Pastikan pada terminal pathnya sudah benar.

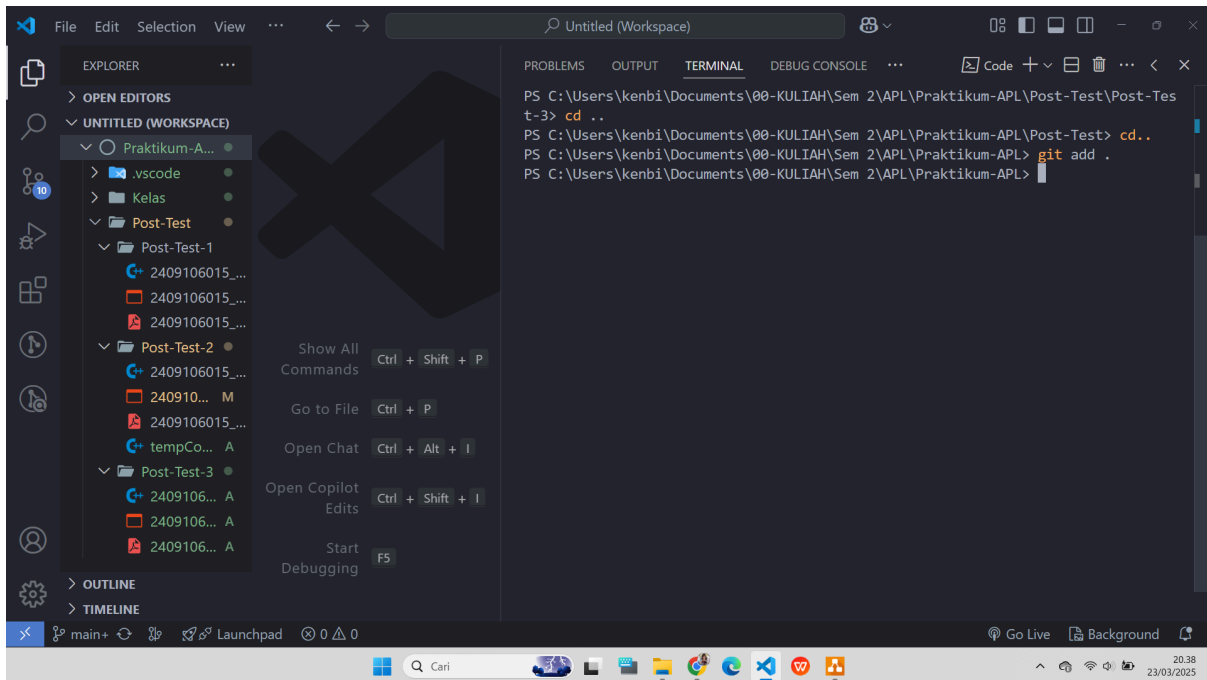
4. Git Init



Gambar 5.8 Git Init

Ketik 'git init' pada terminal, untuk menginisiasi repository git

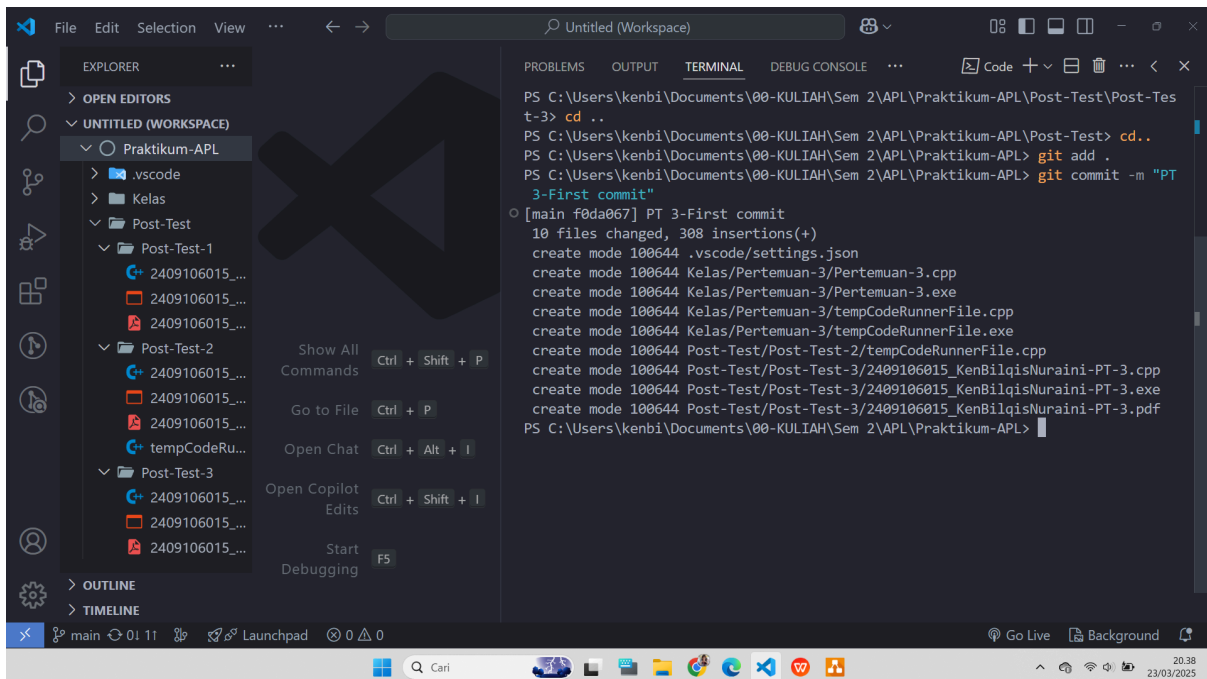
5. Git Add



Gambar 5.9 Git Add

Ketik 'git add .' untuk menambahkan semua isi folder ke repository.

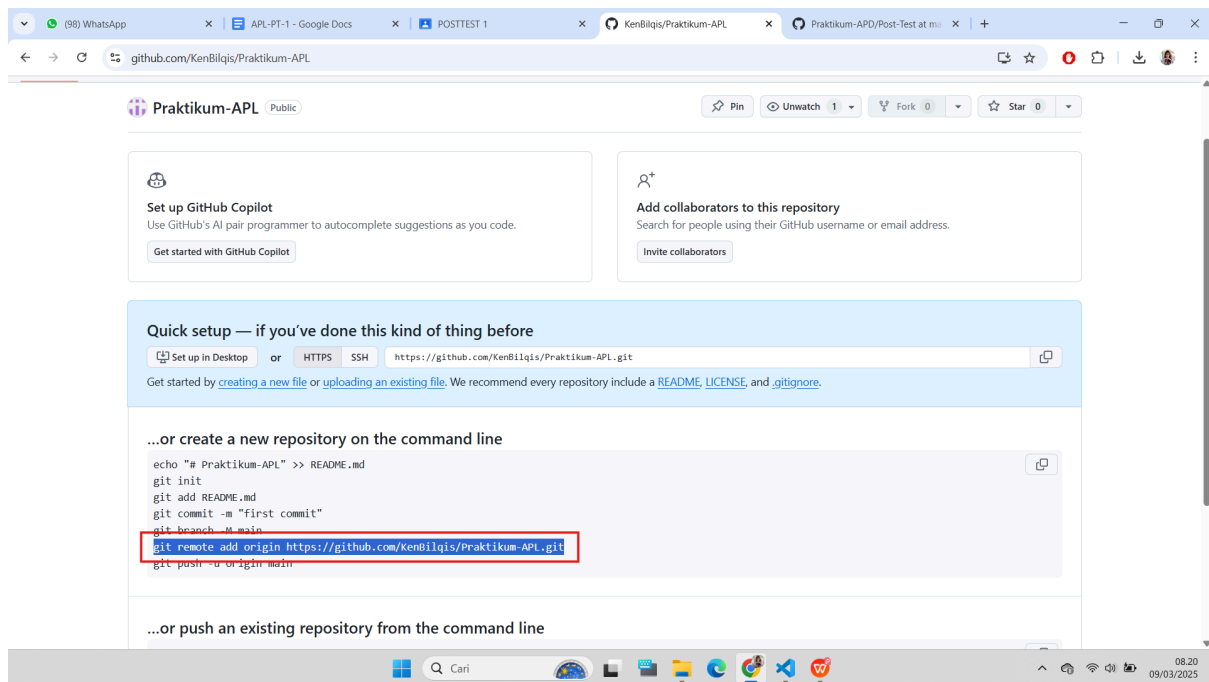
6. Git Commit



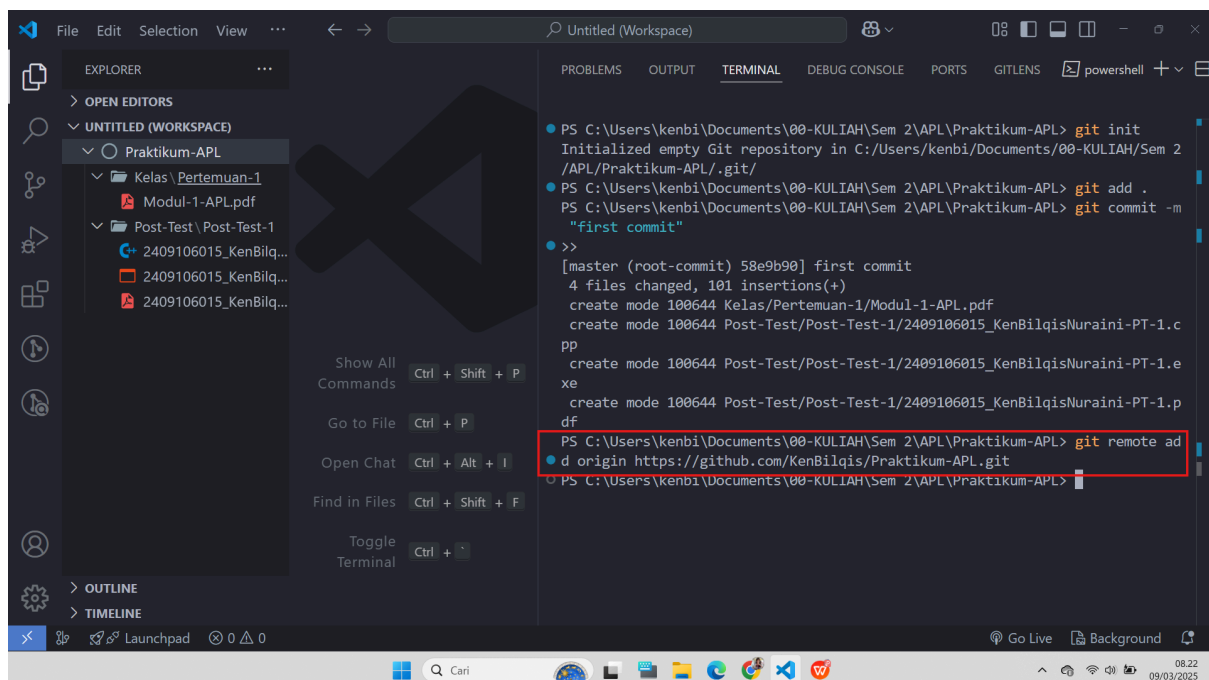
Gambar 5.10 Git Commit

Ketik 'git commit -m "[nama commit]"', untuk membuat semacam checkpoint pada repository

7. Git Remote (Untuk pertama kali remote)



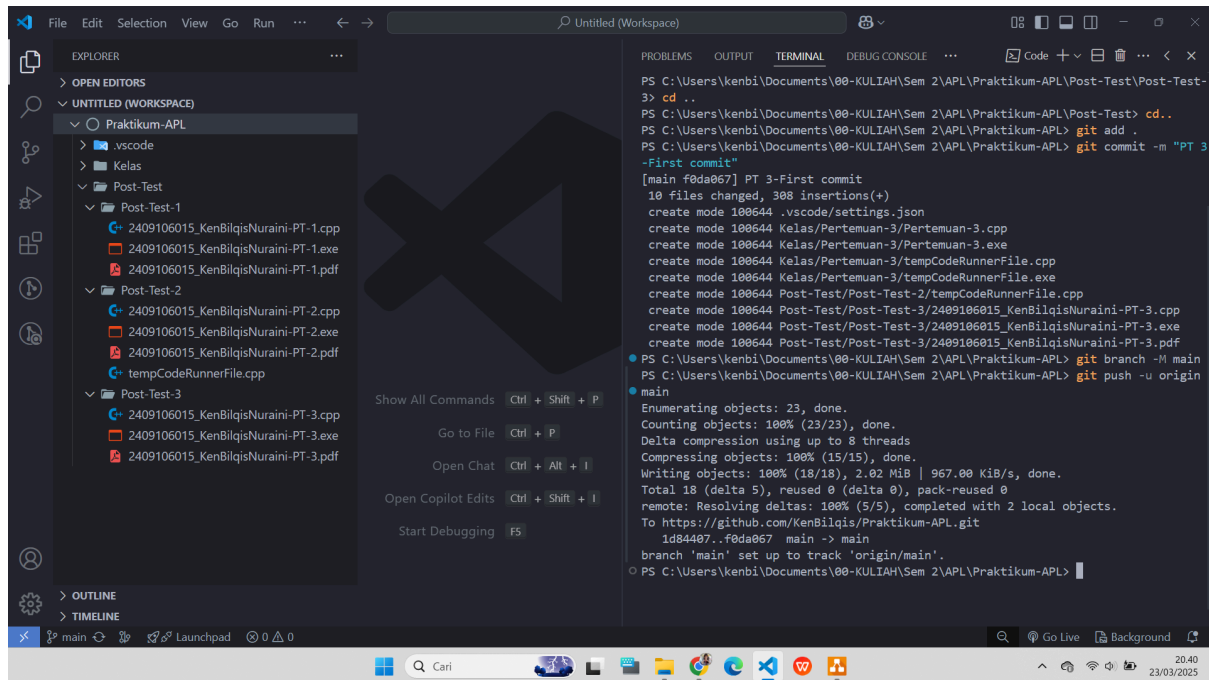
Gambar 5.11 Git Remote di Web



Gambar 5.12 Git Remote

Salin tulisan pada tab sebelumnya seperti pada gambar 5.11, lalu salin di terminal seperti gambar 5.12. Untuk menghubungkan file dari explorer kita ke cloud git

8. Git Push

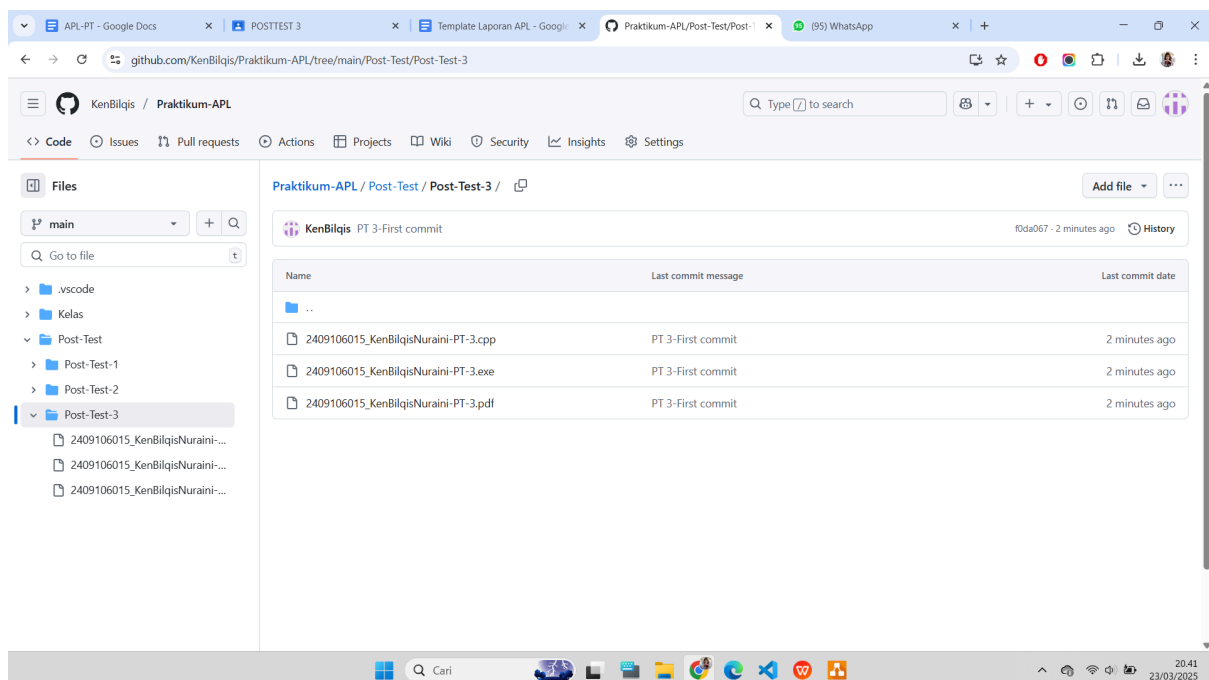


The screenshot shows the VS Code interface with a terminal window open. The terminal displays the following commands and output:

```
PS C:\Users\kenbi\Documents\00-KULIAH\Sem 2\APL\Praktikum-APL\Post-Test\Post-Test-3> cd ..
PS C:\Users\kenbi\Documents\00-KULIAH\Sem 2\APL\Praktikum-APL> git add .
PS C:\Users\kenbi\Documents\00-KULIAH\Sem 2\APL\Praktikum-APL> git commit -m "PT 3-First commit"
[main f0da067] PT 3-First commit
10 files changed, 308 insertions(+)
create mode 100644 .vscode/settings.json
create mode 100644 Kelas/Pertemuan-3/Pertemuan-3.cpp
create mode 100644 Kelas/Pertemuan-3/Pertemuan-3.exe
create mode 100644 Kelas/Pertemuan-3/tempCodeRunnerFile.cpp
create mode 100644 Kelas/Pertemuan-3/tempCodeRunnerFile.exe
create mode 100644 Post-Test/Post-Test-2/tempCodeRunnerFile.cpp
create mode 100644 Post-Test/Post-Test-3/2409106015_KenBilqisNuraini-PT-3.cpp
create mode 100644 Post-Test/Post-Test-3/2409106015_KenBilqisNuraini-PT-3.exe
create mode 100644 Post-Test/Post-Test-3/2409106015_KenBilqisNuraini-PT-3.pdf
PS C:\Users\kenbi\Documents\00-KULIAH\Sem 2\APL\Praktikum-APL> git push -u origin main
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (18/18), 2.02 MiB | 967.00 KiB/s, done.
Total 18 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 2 local objects.
To https://github.com/KenBilqis/Praktikum-APL.git
1d84407..f0da067 main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\kenbi\Documents\00-KULIAH\Sem 2\APL\Praktikum-APL>
```

Gambar 5.13 Git Push

Setelah itu jangan lupa ketik ‘git branch -M main’, untuk membuat percabangan utama pada repository. Lalu ketik ‘git push -u origin main’, untuk mengupload semua file tadi ke cloud github.



Gambar 5.14 Refresh

Setelah semuanya selesai kembali ke web browser tadi dan reload/refresh web tersebut, maka semua file tadi sudah ada pada repository tersebut.