

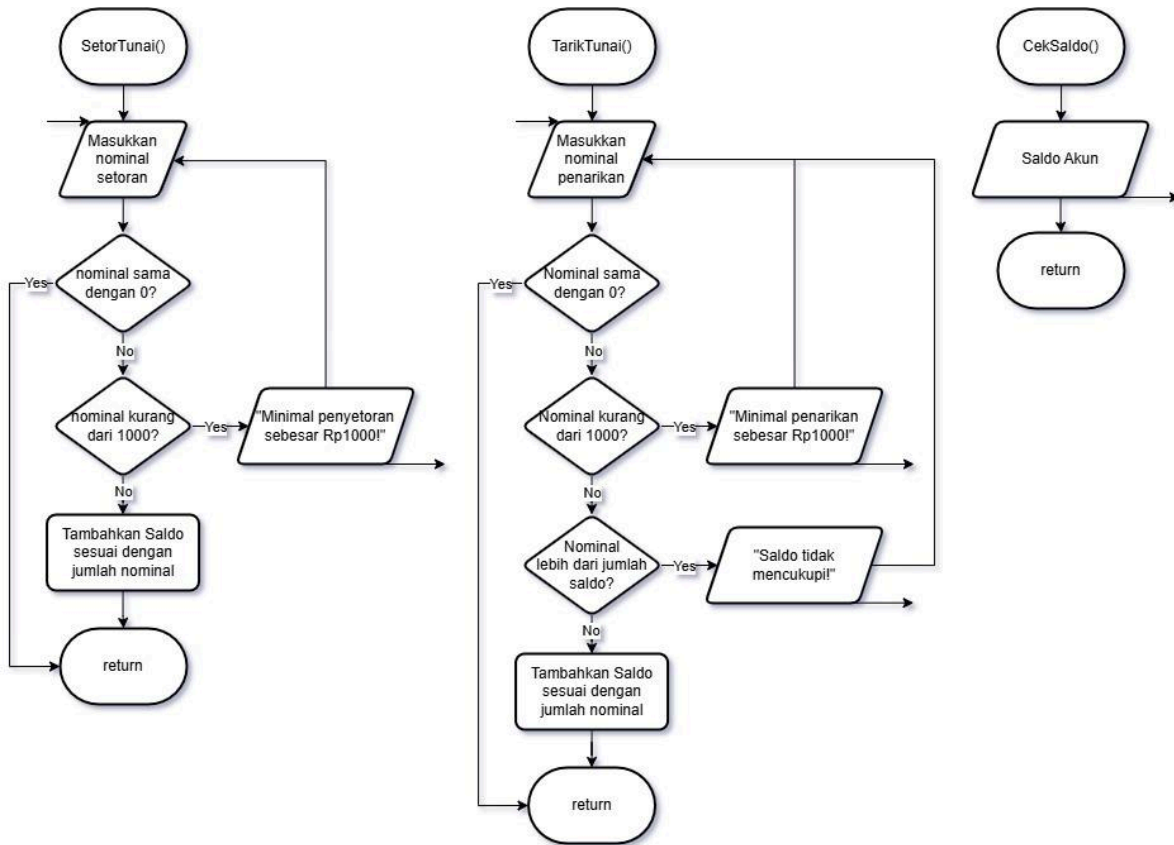
**LAPORAN PRAKTIKUM**  
**POSTTEST 1**  
**ALGORITMA PEMROGRAMAN LANJUT**



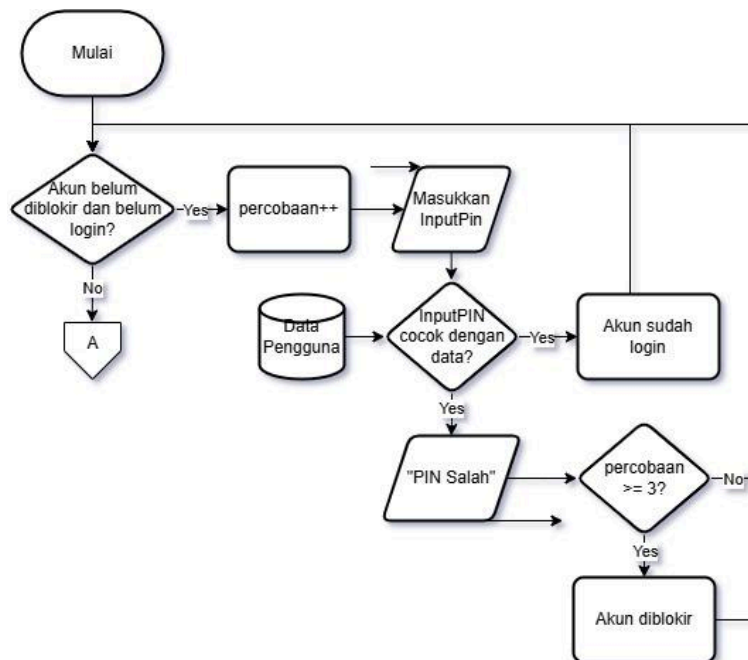
**Disusun oleh:**  
**Nama (2409106060)**  
**Kelas (B1 '24)**

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

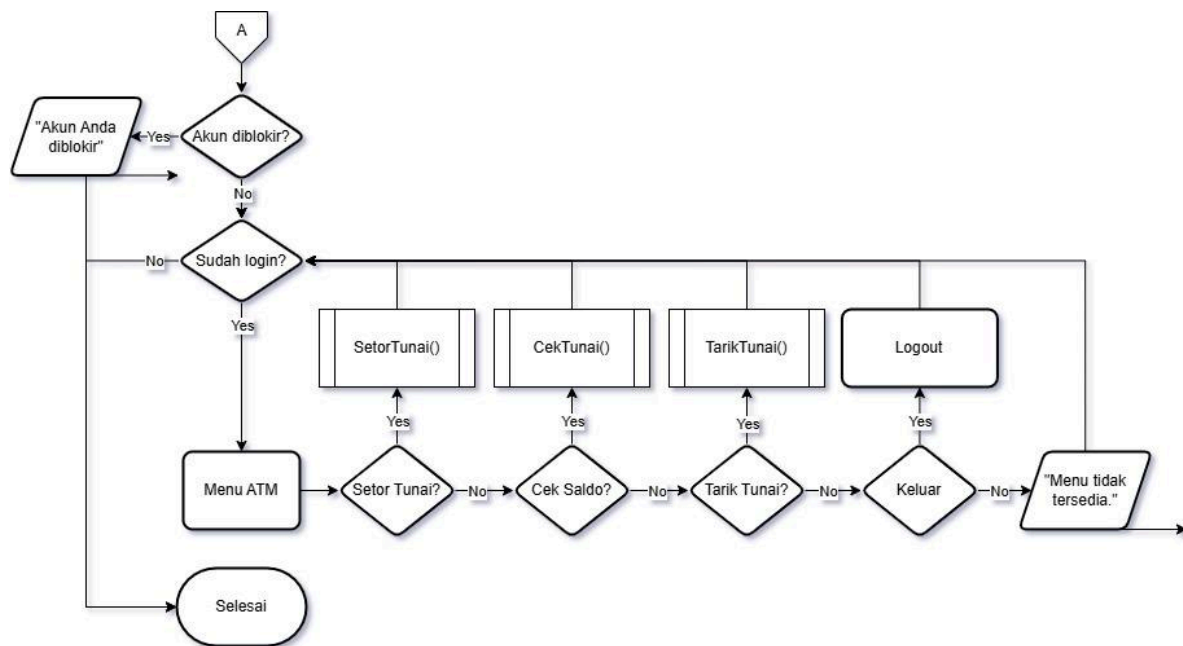
## 1. Flowchart



Gambar 1.1 Flowchart Fungsi *SetorTunai()*, *TarikTunai()*, dan *CekSaldo()*



Gambar 1.2 Main Flow bagian 1



Gambar 1.3 Main Flow bagian 2

## 2. Analisis Program

### 2.1 Deskripsi Singkat Program

Program ATM sederhana ini dibuat bertujuan sebagai sebuah wadah bagi para pengguna untuk mengakses keuangan mereka dengan mudah, fleksibel, dan aman. Di dalam program ini, pengguna dapat menyetorkan, mengecek, dan mengambil saldo rekening sesuai keinginan mereka. Program ini memberikan manfaat kepada para pengguna yang biasanya masih sulit dan lama jika harus mendatangi pihak bank untuk mengelola keuangan mereka, sehingga dengan adanya program ATM ini, diharapkan pengguna dapat merasa lebih mudah dan cepat dalam melakukan transaksi.

### 2.2 Penjelasan Alur & Algoritma

1. Program dimulai setelah berhasil membaca kartu ATM pengguna.
2. Pengguna memasukkan PIN dari kartu ATM yang mereka gunakan.
3. Jika pengguna salah memasukkan PIN, maka akan diberikan pesan bahwa PIN salah, dan jika pengguna salah pada percobaan ketiga, maka percobaan login diberhentikan

dan kartu ATM diblokir. Jika pengguna berhasil memasukkan PIN yang cocok dengan data, maka percobaan login dihentikan dan layar pengguna dipindahkan ke menu ATM dengan pesan bahwa login berhasil.

4. Selama di menu ATM, pengguna dapat memilih menu setor tunai, mengecek saldo, menu tarik tunai, atau keluar dari program ATM.
5. Jika pengguna memilih menu setor tunai, maka pengguna diminta untuk memasukkan nominal setoran yang ingin ditambahkan ke rekening, jika pengguna memasukkan nominal 0, maka pengguna diarahkan kembali ke menu ATM, jika bukan 0 dan pengguna memasukkan nominal kurang dari 1000, maka diberitahu bahwa minimal penyetoran adalah Rp1000, jika pengguna memasukkan nominal sama dengan atau lebih dari 1000, maka diberitahu bahwa saldo berhasil disetorkan dan pengguna diarahkan kembali ke menu utama.
6. Jika pengguna memilih menu cek saldo, maka pengguna diberitahu langsung jumlah saldo yang dimiliki dalam rekening akunnya. Pengguna dapat memilih menu untuk kembali ke menu ATM.
7. Jika pengguna memilih tarik tunai, maka pengguna diminta untuk memasukkan nominal penarikan yang ingin diambil dari rekening, jika pengguna memasukkan nominal 0, maka pengguna diarahkan kembali ke menu ATM, jika bukan 0 dan pengguna memasukkan nominal kurang dari 1000, maka diberitahu bahwa minimal penarikan adalah Rp1000, jika pengguna memasukkan nominal sama dengan atau lebih dari 1000, dan hanya jika nominal penarikan yang dimasukkan lebih dari jumlah saldo yang dimiliki pengguna, maka diberitahu bahwa saldo mencukupi, tetapi jika nominal penarikan kurang dari atau sama dengan 1000, maka diberitahu bahwa saldo berhasil ditarik dan pengguna diarahkan kembali ke menu utama.
8. Jika pengguna memilih menu untuk keluar program, maka program ATM berakhir.

### 3. Source Code

#### A. Sistem Login

Pada bagian awal program, pengguna perlu melalui proses login untuk masuk ke Menu ATM. Data yang harus diberikan dalam proses login adalah PIN. Maksimal percobaan hingga ATM diblokir adalah sebanyak tiga kali.

#### Source Code:

```
// Data Login
string inputPin = "";
int batasPercobaan = 3;
int percobaan = 0;

do
{
    percobaan++;
    cout << "Masukkan PIN: ";
    getline(cin, inputPin);

    if (inputPin != pin)
    {
        // PIN tidak cocok, tampilkan pesan dan cek kelayakan diblokir
        cout << "PIN salah!" << endl;
        isBlokir = percobaan >= batasPercobaan;
    }
    else
    {
        // PIN cocok, tampilkan pesan dan set login untuk ke Menu ATM
        _menuMessage = "Login berhasil!";
        isMenuMessage = true;
        isLogin = true;
    }
} while (!isBlokir && !isLogin);

if (isBlokir)
{
    // Status akun diblokir, tampilkan pesan diblokir
    cout << "Akun Anda diblokir!" << endl;
}
```

#### B. Menu ATM

Bagian berikut digunakan untuk menampilkan menu ATM yang tersedia dan meminta pengguna untuk memilih menu.

### Source Code:

```
cout << "1 > Setor Tunai" << endl;
cout << "2 > Cek Saldo" << endl;
cout << "3 > Tarik Tunai" << endl;
cout << "0 > Keluar" << endl;
cout << endl;
cout << "Pilih menu: ";
cin >> menu;
```

### C. Setor Tunai

Fitur dalam menu setor tunai di sini memungkinkan pengguna untuk menambahkan jumlah saldo pada akun rekening mereka.

### Source Code:

```
double nominalSetor;

while (true)
{
    cout << "======" << endl;
    cout << "Setor Tunai" << endl;
    cout << endl;
    cout << "Masukkan jumlah nominal yang disetorkan (0 > Kembali): " << endl;
    if (isInvalid)
    {
        // Jika sebelumnya melakukan input yang tidak valid, tampilkan pesan
        cout << _invalidMessage << endl;
    }
    cout << endl;
    cout << "Rp";
    cin >> nominalSetor;

    if (nominalSetor == 0)
    {
        // Kembali ke Menu ATM
        break;
    }
    else if (nominalSetor < 1000)
    {
        // Jika nominal kurang dari 1000, tampilkan pesan nominal tidak valid
        _invalidMessage = "Minimal nominal yang bisa disetorkan adalah Rp1000!";
        isInvalid = true;
    }
    else if (nominalSetor >= 1000)
    {
        // Jika nominal lebih dari atau sama dengan 1000, maka tambahkan saldo
```

```

        saldo += nominalSetor;
        _menuMessage = "Saldo berhasil disetorkan!";
        isMenuMessage = true;
        break;
    }
    else
    {
        // Kondisi lainnya jika nominal tidak valid
        _invalidMessage = "Nominal tidak valid!";
        isInvalid = true;
    }
    cout << endl;
}

```

#### D. Cek Saldo

Fitur pada cek saldo berfungsi untuk menampilkan secara langsung jumlah saldo yang dimiliki aku.

##### Source Code:

```

cout << "======" << endl;
cout << "Saldo Rekening Anda: " << endl;
cout << "Rp" << saldo << endl;
cout << endl;
cout << "0 > Kembali: " << endl;
cout << endl;
cout << "Pilih Menu: ";
cin >> temp;

```

#### E. Tarik Tunai

Fitur dalam menu tarik tunai memungkinkan pengguna untuk menarik/mengambil uang melalui saldo yang dimiliki pada akun rekening pengguna.

##### Source Code:

```

double nominalTarik;
bool isInvalid = false;
string _invalidMessage = "";

while (true)
{
    cout << "======" << endl;
    cout << "Tarik Tunai" << endl;
    cout << endl;

```

```

cout << "Masukkan jumlah nominal yang ditarik: " << endl;
if (isInvalid)
{
    // Jika sebelumnya melakukan input yang tidak valid, tampilkan pesan
    cout << _invalidMessage << endl;
}
cout << endl;
cout << "Saldo Rekening Anda: Rp" << saldo << endl;
cout << "Rp";
cin >> nominalTarik;

if (nominalTarik == 0)
{
    // Kembali ke Menu ATM
    break;
}
else if (nominalTarik < 1000)
{
    // Jika nominal kurang dari 1000, tampilkan pesan nominal tidak valid
    _invalidMessage = "Minimal nominal yang bisa ditarik adalah Rp1000!";
    isInvalid = true;
}
else if (nominalTarik >= 1000)
{
    // Jika nominal benar lebih atau sama dengan 1000
    if (nominalTarik <= saldo)
    {
        // Jika saldo mencukupi, kurangi saldo akun dan tampilkan pesan
        saldo -= nominalTarik;
        _menuMessage = "Saldo berhasil ditarik!";
        isMenuMessage = true;
        break;
    }
    else
    {
        // Jika saldo tidak cukup, tampilkan pesan invalid
        _invalidMessage = "Saldo tidak mencukupi!";
        isInvalid = true;
    }
}
else
{
    // Kondisi lainnya jika nominal tidak valid
    _invalidMessage = "Nominal tidak valid!";
    isInvalid = true;
}
cout << endl;
}

```



## 4. Uji Coba dan Hasil Output

### 4.1 Uji Coba

1. Kane yang memiliki simpanan uang tersisa Rp50000 di akun rekeningnya, karena Kane memegang cukup uang tunai, Kane ingin menambah dan menyimpannya sebanyak Rp250000. Kemudian, Kane akan mengecek saldonya kembali untuk memastikan apakah uang saya sudah masuk ke dalam akun saya.
2. Trigusni sedang berbelanja di toko, saat Trigusni ingin membayar melalui *online*, ternyata layanan pembayaran *online* toko tersebut sedang dalam pemeliharaan. Akhirnya, Trigusni mencari ATM terdekat untuk menarik tunai dari saldo yang dia miliki di akun rekeningnya. Andi saat ini memiliki saldo sebesar Rp606060 dan ingin menariknya sebesar Rp250000 dari saldonya. Setelah itu, dia mengecek kembali jumlah saldo yang dia miliki setelah ditariknya.
3. Hermawan yang tidak begitu mengingat PIN-nya gagal sebanyak lebih dari tiga kali dan mendapati akunnya diblokir.

### 4.2 Hasil Output

#### A. Skenario 1 (Menyetor Tunai)

```
Masukkan PIN: 6060
=====
Login berhasil!
```

Gambar 4.1 Proses Login berhasil (Skenario 1)

```
1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu: █
```

Gambar 4.2 Menu ATM setelah berhasil login (Skenario 1)

```

=====
Saldo Rekening Anda:
Rp50000

0 > Kembali:

Pilih Menu: 0

=====

```

Gambar 4.3 Menu Cek Saldo sebelum disetorkan (Skenario 1)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu: █

```

Gambar 4.4 Kembali ke Menu ATM dari Menu Cek Saldo (Skenario 1)

```

=====
Setor Tunai

Masukkan jumlah nominal yang disetorkan (0 > Kembali):

Rp250000

=====
Saldo berhasil disetorkan!

```

Gambar 4.5 Menu Setor Tunai (Skenario 1)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu: █

```

Gambar 4.6 Kembali ke Menu ATM setelah berhasil menyetorkan saldo (Skenario 1)

```

=====
Saldo Rekening Anda:
Rp300000

0 > Kembali:

Pilih Menu: 0

=====

```

Gambar 4.6 Menu Cek Saldo setelah penyetoran saldo (Skenario 1)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu: 0

Keluar dari program...
PS D:\Kuliah (Git Repository)\Praktikum-APL\praktikum-apl>

```

Gambar 4.7 Kembali ke Menu Utama dan keluar dari program ATM (Skenario 1)

## B. Skenario 2 (Menarik Tunai)

```

Masukkan PIN: 6060

=====
Login berhasil!

```

Gambar 4.8 Proses berhasil login (Skenario 2)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu:

```

Gambar 4.9 Menu ATM setelah berhasil login (Skenario 2)

```

=====
Saldo Rekening Anda:
Rp606060

0 > Kembali:

Pilih Menu: 0

=====

```

Gambar 4.10 Menu Cek Saldo sebelum menarik tunai (Skenario 2)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu:

```

Gambar 4.10 Kembali ke Menu ATM dari Cek Saldo dan ingin menarik tunai (Skenario 2)

```

=====
Tarik Tunai

Masukkan jumlah nominal yang ditarik:

Saldo Rekening Anda: Rp606060
Rp250000

=====
Saldo berhasil ditarik!

```

Gambar 4.11 Menu Tarik Tunai (Skenario 2)

```

1 > Setor Tunai
2 > Cek Saldo
3 > Tarik Tunai
0 > Keluar

Pilih menu: 

```

Gambar 4.12 Kembali ke Menu ATM setelah berhasil menarik saldo (Skenario 2)

```

=====
Saldo Rekening Anda:
Rp356060

0 > Kembali:

Pilih Menu: 0

=====

```

Gambar 4.12 Menu Cek Saldo setelah penarikan tunai (Skenario 2)

### C. Skenario 3 (Akun Terblokir)

```

Masukkan PIN: 1234
PIN salah!

```

Gambar 4.13 Memasukkan PIN yang salah pada percobaan pertama (Skenario 3)

```

Masukkan PIN: admin123
PIN salah!

```

Gambar 4.14 Memasukkan kembali PIN yang salah pada percobaan kedua (Skenario 1)

```

Masukkan PIN: user
PIN salah!
Akun Anda diblokir!
PS D:\Kuliah (Git Repository)\Praktikum-APL\praktikum-apl> 

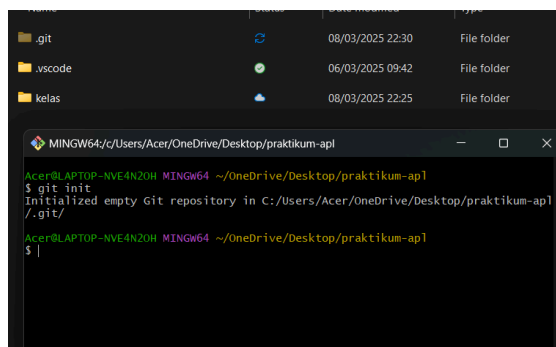
```

Gambar 4.15 Akun diblokir setelah salah memasukkan PIN sebanyak 3 kali (Skenario 3)

## 4. Uji Coba dan Hasil Output

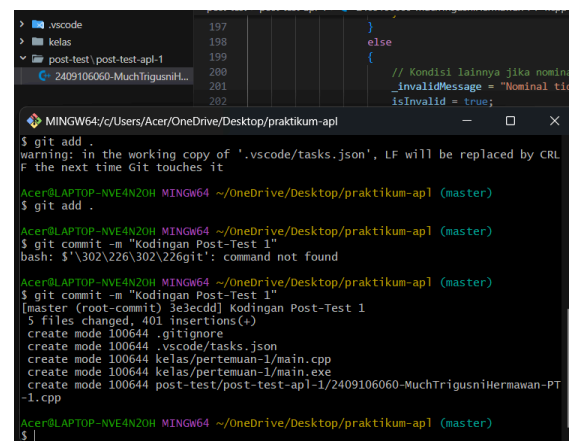
### A. Membuat Repository di GitHub

1. *git init*, perintah ini digunakan pada saat pertama kali membuat *repository* di direktori.
2. *git add*, perintah ini digunakan untuk menambahkan *file* yang ingin dimasukkan ke dalam *commit* berikutnya.
3. *git commit*, setelah melakukan *git add*, perintah ini digunakan untuk membuat semacam *checkpoint* file yang terdapat ditambahkan pada *git add*, sehingga kondisi atau data file pada waktu ini dapat dilihat kembali di masa depan.
4. *git remote*, perintah ini digunakan untuk menghubungkan dengan layanan git server seperti github, sehingga repository *local* yang terdapat pada *local machine* kita dapat disambungkan/disinkronisasikan dengan repository di github.
5. *git push*, perintah ini digunakan jika sebelumnya telah melakukan *git commit* dan telah terhubung dengan *repository* di server, commit di repository *git local* akan di-upload ke server di branch terkait.



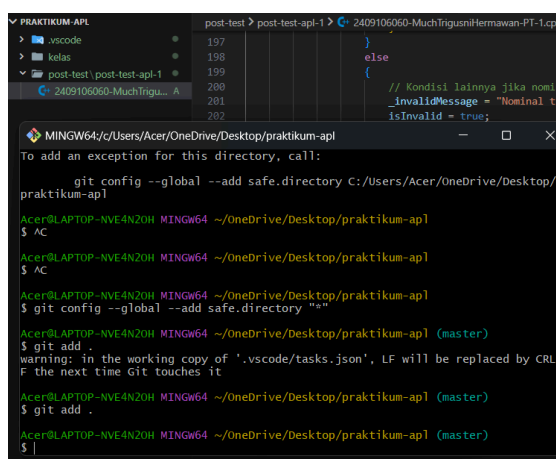
```
MINGW64/c/Users/Acer/OneDrive/Desktop/praktikum-apl
$ git init
Initialized empty Git repository in C:/Users/Acer/OneDrive/Desktop/praktikum-apl/.git/
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl
$ |
```

Gambar 4.1 *git init*



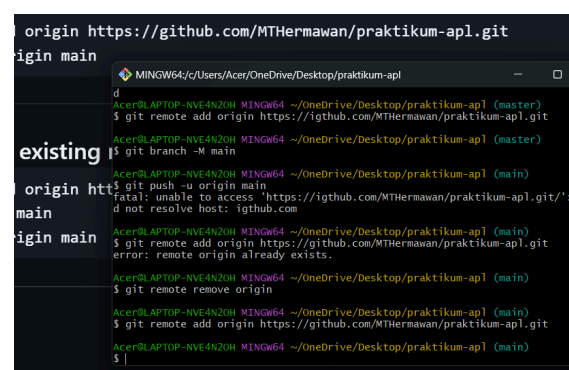
```
MINGW64/c/Users/Acer/OneDrive/Desktop/praktikum-apl
$ git add
warning: in the working copy of '.vscode/tasks.json', LF will be replaced by CRLF the next time Git touches it
$ git add .
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ git commit -m "Kodingan Post-Test 1"
bash: '$\302\226\302\226git': command not found
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ git commit -m "Kodingan Post-Test 1"
[master (root-commit) 3e3ecd] Kodingan Post-Test 1
5 files changed, 401 insertions(+)
create mode 100644 .gitignore
create mode 100644 .vscode/tasks.json
create mode 100644 kelas/pertemuan-1/main.cpp
create mode 100644 kelas/pertemuan-1/main.exe
create mode 100644 post-test/post-test-apl-1/2409106060-MuchTrigusniHermawan-PT-1.cpp
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ |
```

Gambar 4.3 *git commit*



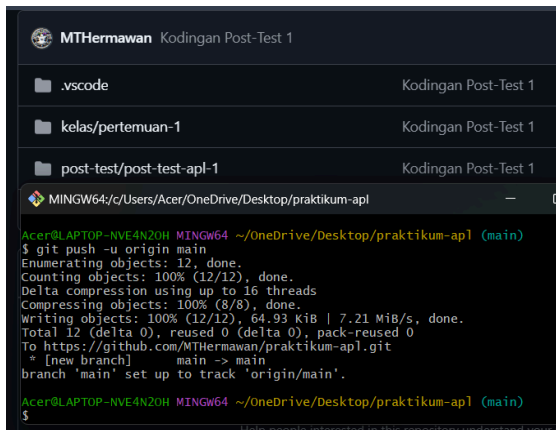
```
PRAKTIKUM-APL
$ git add .
warning: in the working copy of '.vscode/tasks.json', LF will be replaced by CRLF the next time Git touches it
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ git add .
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ |
```

Gambar 4.2 *git add*



```
origin https://github.com/MTHermawan/praktikum-apl.git
$ git remote add origin https://github.com/MTHermawan/praktikum-apl.git
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (master)
$ git push -u origin main
fatal: unable to access 'https://github.com/MTHermawan/praktikum-apl.git/':
d not resolve host: github.com
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (main)
$ git remote remove origin
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (main)
$ git remote add origin https://github.com/MTHermawan/praktikum-apl.git
Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (main)
$ |
```

Gambar 4.4 *git remote*



```
MTHermawan Kodingan Post-Test 1

.vscode Kodingan Post-Test 1
kelas/pertemuan-1 Kodingan Post-Test 1
post-test/post-test-apl-1 Kodingan Post-Test 1
MINGW64/c/Users/Acer/OneDrive/Desktop/praktikum-apl

Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (main)
$ git push -u origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 16 threads
Compressing objects: 100% (8/8), done.
writing objects: 100% (12/12), 64.93 KiB | 7.21 MiB/s, done.
Total 12 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MTHermawan/praktikum-apl.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

Acer@LAPTOP-NVE4N20H MINGW64 ~/OneDrive/Desktop/praktikum-apl (main)
$
```

Gambar 4.5 *git push*