

**LAPORAN TUGAS MINI PROJECT PEMROGRAMAN
BERORIENTASI OBJEK
“TEXT EDITOR”**

[MiniProjectPBO](#)



Dosen Pengampu:

Prof. Dr. Drs. Opim Salim Sitompul M.Sc

Reza Taqyuddin S.Kom

Disusun Oleh :

KELOMPOK 5

Ikhwan Prananta Hasugian (221402004)

Ilyas Al Muadz (221402029)

Jeconiah S.C Nababan (221402069)

FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

UNIVERSITAS SUMATERA UTARA MEDAN

TAHUN AJARAN 2022/2023

KATA PENGANTAR

Puji dan syukur senantiasa kita panjatkan kepada Tuhan Yang Maha Esa dengan rahmat dan karunia-Nya sehingga kami dapat menyelesaikan penyusunan laporan “Text Editor” kami dalam rangka pemenuhan tugas mini project mata kuliah Pemrograman Berorientasi Objek.

Tidak lupa penulis mengucapkan rasa terima kasih kepada Prof. Dr. Drs. Opim Salim Sitompul M.Sc dan Reza Taqyuddin S.Kom selaku dosen pengampu dan asisten dosen untuk mata kuliah pemrograman berorientasi objek yang telah membantu penulis dalam mengerjakan laporan ini. Penulis juga mengucapkan terima kasih kepada teman-teman yang telah memberikan masukan dalam pembuatan laporan ini.

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna dan masih terdapat beberapa kekurangan, oleh karena itu penulis sangat mengharapkan saran dan kritik yang membangun dari pembaca untuk penyempurnaan laporan ini. Semoga laporan mini project ini dapat bermanfaat bagi pembaca dan juga untuk penulis sendiri.

Medan, 09 Juni 2023

Penulis

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI	ii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Tujuan.....	2
1.3 Hasil yang diharapkan	2
BAB II TAMPILAN MENU.....	3
2.1 Tampilan Menu Program	3
BAB III HASIL IMPLEMENTASI DAN SCREESHOT TAMPILAN	4
3.1 Fitur Create New File	4
3.2 Fitur Read and Open File	4
3.3 Fitur Edit a Line From File.....	5
3.4 Fitur Delete File.....	7
3.5 Fitur Compile Code (only C++)	8
3.6 Fitur Run File	9
3.7 Tampilan code	9
3.7.1 Main Code	9
3.7.2 Class Code.....	11
BAB IV PENUTUP	16
4.1 Kesimpulan.....	16
4.2 Saran	16

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam era perkembangan teknologi informasi yang pesat, penggunaan text editor telah menjadi bagian penting dalam kegiatan pengembangan perangkat lunak dan penulisan kode program. Text editor adalah salah satu alat yang digunakan untuk mengedit dan mengelola file teks. Namun, dalam industri perangkat lunak, terdapat berbagai macam text editor yang tersedia dengan fitur dan fungsionalitas yang berbeda-beda.

Salah satu text editor populer yang telah dikenal di kalangan pengembang perangkat lunak adalah Vim. Vim adalah text editor yang berada di dalam terminal dan memiliki penggunaan serta konfigurasi yang cukup berbeda dari text editor lainnya. Vim memungkinkan pengguna untuk melakukan pengeditan teks dengan efisiensi tinggi melalui berbagai perintah dan pintasan keyboard yang canggih.

Dalam konteks ini, kami memilih untuk membuat text editor sederhana sebagai tugas mini proyek dari mata kuliah pemrograman berorientasi objek. Text Editor ini bertujuan untuk memberikan pengalaman pengeditan teks yang intuitif, efisien, dan fleksibel, dengan menggunakan prinsip-prinsip desain dan konsep pemrograman berorientasi objek.

1.2 Tujuan

Tujuan dari proyek Text Editor ini adalah:

1. Mengembangkan text editor yang memiliki antarmuka pengguna yang intuitif dan mudah digunakan.
2. Menyediakan fitur-fitur yang mendukung pengeditan teks yang efisien dan produktif.
3. Meningkatkan pemahaman dan penerapan konsep pemrograman berorientasi objek dalam pengembangan perangkat lunak.
4. Memenuhi tugas mini project yang diberikan sebagai tugas project akhir mata kuliah “Pemrograman Berorientasi Object”.

1.3 Hasil yang diharapkan

Dengan dikembangkannya Text Editor kami, diharapkan akan memberikan manfaat sebagai berikut:

1. Meningkatkan efisiensi dan produktivitas pengguna dalam pengeditan teks dan pengembangan perangkat lunak.
2. Menyediakan pengalaman pengeditan teks yang intuitif dan sesuai dengan kebutuhan pengguna.
3. Memperkaya pemahaman pengembang tentang konsep-konsep pemrograman berorientasi objek.
4. Mendapatkan nilai yang baik dari hasil kerja keras kami.

Dengan demikian, penelitian ini memiliki nilai penting dalam bidang pengembangan perangkat lunak dan konsep pemrograman berorientasi objek dan diharapkan pengguna dapat mengoptimalkan proses pengeditan teks dan meningkatkan efisiensi dalam pengembangan perangkat lunak.

BAB III

TAMPILAN MENU

2.1 Tampilan Menu Program

```
D:\MiniProjectPBO>.\test
Masukkan Nama File: projectpbo.cpp

--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: |
```

Di dalam Tampilan Menu tersebut mencakup delapan fitur yang terdiri dari:

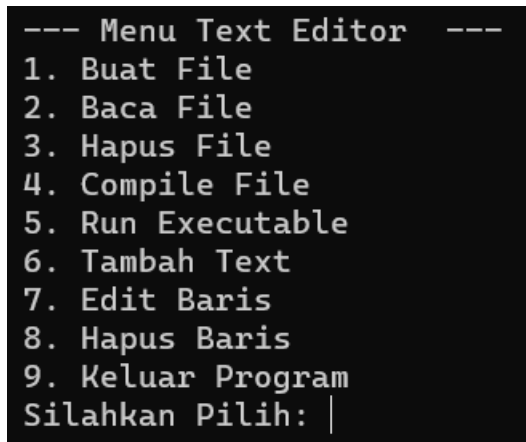
- Create New File
- Read File
- Delete File
- Add Line
- Edit Line
- Delete File
- Compile Code (only C++)
- Run File File

BAB IV

HASIL IMPLEMENTASI DAN SCREESHOOT TAMPILAN

3.1 Fitur Create File

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Buat File" yang memungkinkan pengguna untuk membuat file baru dengan ekstensi yang sesuai. Pengguna dapat dengan mudah membuat file baru dengan format yang diinginkan. Perhatikan *Gambar 1.1*.



```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: |
```

3.2 Fitur Read File

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Baca File" yang memungkinkan pengguna untuk membaca dan membuka file yang telah dibuat sebelumnya. Dengan menggunakan fitur "Baca File", pengguna harus menulis nama file beserta ekstensi yang telah dibuat sebelumnya. Perhatikan *Gambar 2.1*.

```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 2
Isi File:
include <iostream>

using namespace stdl

int main(){

cout<<"hello world";

return 0;
}
```

3.3 Fitur Edit Line

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Edit Baris" yang memberikan kemampuan kepada pengguna untuk melakukan perubahan atau modifikasi pada baris-baris yang terdapat dalam file mereka. Perhatikan *Gambar 3.1*.


```

--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 7
Masukkan Nomor Baris Yang Ingin Di Edit: 3
Masukkan Baris Yang Baru: using namespace std;
Baris Yang Diedit: using namespace stdl Menjadi: using namespace std;

```

Fitur "Edit Line " memiliki tiga fitur tambahan yaitu:

- "Tambah Text" yang berguna untuk menambahkan baris baru dalam file. pengguna dapat langsung memasukkan teks atau kode yang ingin ditambahkan pada file tersebut. Perhatikan *Gambar 3.2.1*.

```

--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 6
Tuliskan Yang Ingin Anda Tulis (Ketik'seleasai' untuk berhenti):
include<iostream>

using namespace std;

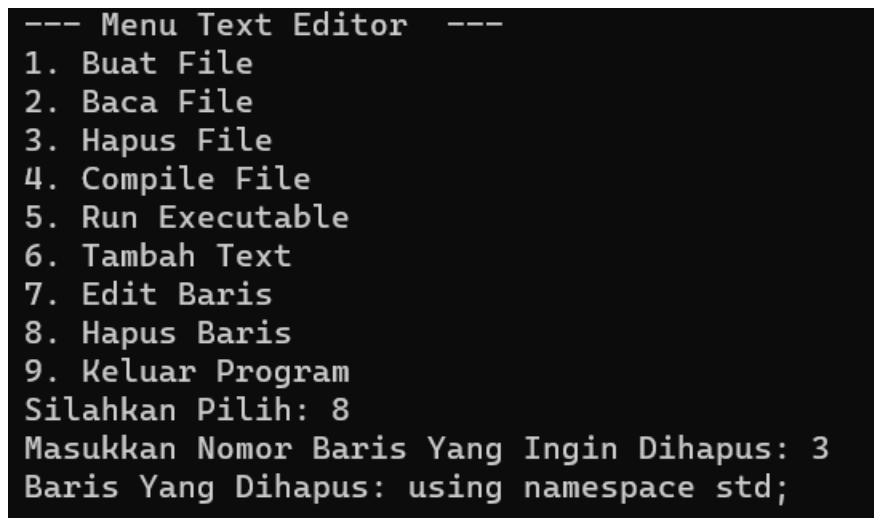
int main(){

cout<<"Hello World";

return 0;
}
selesai

```

- "Hapus Baris" yang berguna untuk menghapus baris yang tidak diinginkan dalam file. Fitur ini memberikan pengguna kemampuan untuk dengan mudah menghilangkan baris-baris tertentu yang tidak. Untuk menggunakan fitur ini, pengguna hanya perlu memilih nomor baris yang ingin dihapus dari daftar baris yang ada. Dengan menentukan nomor baris tersebut, pengguna dapat secara langsung menghapus baris tersebut dari file. Perhatikan *Gambar 3.4.1*.



```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 8
Masukkan Nomor Baris Yang Ingin Dihapus: 3
Baris Yang Dihapus: using namespace std;
```

Dengan adanya fitur pengeditan ini, pengguna memiliki kontrol penuh atas isi file mereka. Mereka dapat menyesuaikan dan mengubah konten file dengan mudah sesuai dengan kebutuhan proyek atau tugas yang sedang dijalankan.

3.4 Fitur Delete File

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Delete File" yang memberikan kemampuan kepada pengguna untuk menghapus file yang sudah dibuat. Perhatikan *Gambar 4.1*.

```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 3
File Berhasil Dihapus: projectpbo.cpp
```

Setelah di-ENTER akan muncul tampilan berikut. Perhatikan *Gambar 4.2*.

3.5 Fitur Compile Code (only C++)

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Compile Code (only c++)" yang secara khusus dirancang untuk file dengan ekstensi (.cpp). Fitur ini memberikan kemampuan kepada pengguna untuk mengompilasi file kode program dalam bahasa C++ menjadi file objek yang dapat dieksekusi. Perhatikan *Gambar 5.1*.

```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 4
Compile Berhasil..
```

3.6 Fitur Run File

Dalam Tampilan Menu proyek Text Editor kami, terdapat fitur "Run File" yang memungkinkan pengguna untuk menjalankan berkas yang telah di-compile menjadi file eksekusi (.exe). Perhatikan *Gambar 6.1*.

```
--- Menu Text Editor ---
1. Buat File
2. Baca File
3. Hapus File
4. Compile File
5. Run Executable
6. Tambah Text
7. Edit Baris
8. Hapus Baris
9. Keluar Program
Silahkan Pilih: 5
Hello WorldExecuting Program Berhasil.
```

3.7 Tampilan code

3.7.1 Main Code

```
int main()
{
    string filename;

    //Meminta User Untuk Menginput Nama File
    cout << "Masukkan Nama File: ";
    getline(cin, filename); //Membaca nama file yang dimasukkan oleh pengguna.

    TextEditor textEditor(filename); //Membuat objek 'TextEditor' dengan
menggunakan nama file yang diberikan.

    int choice;
    // Melakukan perulangan berdasarkan pilihan pengguna sampai pilihan 9
(keluar) dipilih.
    do {
        cout << "\n--- Menu Text Editor ---" << endl;
        cout << "1. Buat File" << endl;
        cout << "2. Baca File" << endl;
        cout << "3. Hapus File" << endl;
```

```

    cout << "4. Compile File" << endl;
    cout << "5. Run Executable" << endl;
    cout << "6. Tambah Text" << endl;
    cout << "7. Edit Baris" << endl;
    cout << "8. Hapus Baris" << endl;
    cout << "9. Keluar Program" << endl;
    cout << "Silahkan Pilih: ";
    cin >> choice; //Membaca pilihan yang dimasukkan oleh pengguna.
    cin.ignore(); //Mengabaikan karakter newline (\n) setelah membaca pilihan
    agar tidak terbaca sebagai masukan berikutnya.

    //Memeriksa pilihan pengguna dan menjalankan fungsi yang sesuai
    berdasarkan pilihan user.
    switch (choice)
    {
        case 1:
            textEditor.createFile();
            break;
        case 2:
            textEditor.readFile();
            break;
        case 3:
            textEditor.deleteFile();
            break;
        case 4:
            textEditor.compileFile();
            break;
        case 5:
            textEditor.runExecutable();
            break;
        case 6:
        {
            string line;
            cout << "Tuliskan Yang Ingin Anda Tulis (Ketik'seleasai' untuk
berhenti): " << endl;
            getline(cin, line);
            textEditor.addLine(line);
            break;
        }
        case 7:
        {
            int lineNumber;
            string newLine;
            cout << "Masukkan Nomor Baris Yang Ingin Di Edit: ";
            cin >> lineNumber; //Meminta User Untuk Menginput Nomor Baris

```

```

        cin.ignore();
        cout << "Masukkan Baris Yang Baru: ";
        getline(cin, newLine); //Meminta User Untuk Menginput Baris Baru
        textEditor.editLine(lineNumber, newLine);
        break;
    }
    case 8:
    {
        int lineNumber;
        cout << "Masukkan Nomor Baris Yang Ingin Dihapus: ";
        cin >> lineNumber;
        cin.ignore();
        textEditor.deleteLine(lineNumber);
        break;
    }
    case 9:
        cout << "Keluar Dari Program..." << endl;
        break;
    default:
        cout << "Pilihan Tidak Sesuai,Coba Lagi..." << endl;
    }
} while (choice != 9);

return 0;
}

```

3.7.2 Class Code

```

class TextEditor
{
    private:
        string filename; //Mendeklarasikan variabel 'filename' sebagai string
        untuk menyimpan nama file.

    public:
        //Constuctor
        TextEditor(string fname)
        {
            filename = fname;
        }

        //Fungsi Untuk Membuat File
        void createFile()

```

```

{
    ofstream file(filename.c_str()); //Membuka File
    file.close();//Menutup File
    cout << "File Berhasil Dibuat: " << filename << endl;
}

//Fungsi untuk membaca isi file yang telah dibuat dan menampilkannya di
layar.
void readFile()
{
    ifstream file(filename.c_str());//Membuka File
    string line;
    cout << "Isi File: " << endl;
    while (getline(file, line))
    {
        cout << line << endl;
    }
    file.close(); //Menutup File
}

//Fungsi untuk menghapus file yang telah dibuat.
void deleteFile()
{
    if (remove(filename.c_str()) != 0) //Menghapus file dengan
menggunakan 'remove()' dan mengembalikan pesan kesalahan jika terjadi kegagalan.
    {
        cerr << "Error Menghapus File." << endl;
    }
    else
    {
        cout << "File Berhasil Dihapus: " << filename << endl;
    }
}

//Fungsi untuk mengompilasi file dengan menggunakan kompiler 'g++' dan
menghasilkan file eksekusi dengan ekstensi '.exe'.
void compileFile()
{
    // Membentuk perintah kompilasi dengan menggunakan nama file yang
diberikan.
    string command = "g++ " + filename + " -o " +
getFileNameWithoutExtension() + ".exe";
    if (system(command.c_str()) == 0) //Menjalankan perintah kompilasi
menggunakan 'system()' dan memeriksa apakah kompilasi berhasil.
    {

```

```

        cout << "Compile Berhasil.." << endl;
    }
    else
    {
        cerr << "Compile Gagal.." << endl;
    }
}

//Fungsi untuk menjalankan file eksekusi yang telah dihasilkan.
void runExecutable()
{
    //Membentuk perintah untuk menjalankan file eksekusi.
    string command = getFileNameWithoutExtension() + ".exe";
    if (system(command.c_str()) == 0)
    {
        cout << "Executing Program Berhasil." << endl;
    }
    else
    {
        cerr << "Executing Program Gagal." << endl;
    }
}

//Fungsi untuk menambahkan baris ke dalam file.
void addLine(string line)
{
    ofstream file(filename.c_str(), ios::app);
    while (line != "selesai") //Looping Penulisan Kalimat Dan Berhenti
Ketika User Menuliskan Kata 'Selesai'
    {
        file << line << endl;
        getline(cin, line);
    }
    file.close(); //Menutup File
}

//Fungsi untuk mengedit baris tertentu dalam file dengan mengganti baris
tersebut dengan 'newLine'.
void editLine(int lineNumber, string newLine)
{
    ifstream inputFile(filename.c_str());
    ofstream outputFile("temp.txt");
    string line;
    int currentLine = 1;

```



```

        while (getline(inputFile, line))
        {
            if (currentLine == lineNumber)
            {
                outputFile << newLine << endl;
                cout << "Baris Yang Diedit: " << line << " Menjadi: " <<
newLine << endl;
            }
            else
            {
                outputFile << line << endl;
            }
            currentLine++;
        }

        inputFile.close();
        outputFile.close();

        remove(filename.c_str());
        rename("temp.txt", filename.c_str());
    }

//Fungsi Untuk Menghapus Baris Pada File
void deleteLine(int lineNumber)
{
    ifstream inputFile(filename.c_str());
    ofstream outputFile("temp.txt");
    string line;
    int currentLine = 1;

    while (getline(inputFile, line))
    {
        if (currentLine != lineNumber)
        {
            outputFile << line << endl;
        } else
        {
            cout << "Baris Yang Dihapus: " << line << endl;
        }
        currentLine++;
    }

    inputFile.close();
    outputFile.close();
}

```

```
        remove(filename.c_str());  
        rename("temp.txt", filename.c_str());  
    }  
  
private:  
    //Fungsi privat untuk mendapatkan nama file tanpa ekstensi.  
    string getFileNameWithoutExtension()  
    {  
        size_t pos = filename.find_last_of(".");  
        return filename.substr(0, pos);  
    }  
};
```

Untuk penjelasan code yang lebih detail, bisa dilihat di link GitHub berikut :

[MiniProjectPBO](#)

BAB V

PENUTUP

4.1 Kesimpulan

Kesimpulan dari project Text editor ini ialah sebagai berikut :

- Pengguna dapat menggunakan Text Editor ini untuk membuat sebuah file dengan ekstensi tersendiri.
- Pengguna dapat menggunakan Text Editor untuk menulis sebuah konten / isi kedalam sebuah file, dan juga membaca isi file tersebut langsung dari terminal.
- Pengguna dapat menggunakan aplikasi ini untuk melakukan edit dalam sebuah file tertentu langsung di terminal, yang dimana pengguna dapat melakukan perubahan, maupun menghapus sebuah baris langsung dari terminal.
- Pengguna dapat menggunakan aplikasi ini untuk melakukan perintah compile sebuah program C++ langsung dari terminal.
- Pengguna dapat menggunakan aplikasi ini untuk menjalankan sebuah aplikasi yang telah di compile maupun aplikasi yang telah tersedia.
- Aplikasi ini telah memenuhi persyaratan dan memenuhi ketentuan tugas yang diberikan dalam mata kuliah “Pemrograman Berorientasi Objek”.

4.2 Saran

Saran untuk project Text Editor adalah sebagai berikut :

- Developer butuh mengatasi penggunaan memori berlebihan dari aplikasi yang disebabkan oleh ketidakefisienan aplikasi.
- Developer butuh memberikan desain interaksi yang lebih menarik dan juga sepadan untuk pengguna nantinya.
- User harus mengusahakan memahami dokumentasi secara mendalam agar terbiasa dalam penggunaan aplikasi ini.