

IMPLEMENTASI STRUKTUR DATA PADA APLIKASI PENGADUAN WARGA ONLINE

Implementation of Data Structure in Online Citizen Complaint Application

NISA AMELIA (G6401231022), MOHAMMAD MIRZA SHAHBAZ AVIANTO (G6401231143), LATHIIFA ZAHIRA YAHYA (G6401231154)

Abstrak

Pelayanan publik berbasis digital semakin dibutuhkan untuk meningkatkan efisiensi dan transparansi, salah satunya dalam hal pengaduan warga. Proyek ini bertujuan mengembangkan aplikasi pengaduan warga online berbasis terminal menggunakan bahasa pemrograman C++ dengan penerapan berbagai struktur data. Struktur data seperti queue digunakan untuk mengelola antrean laporan, stack untuk fitur undo, hashing untuk menghasilkan ID laporan yang unik, serta enkripsi untuk menjaga privasi data pelapor. Sorting digunakan untuk menyusun laporan berdasarkan tingkat prioritas. Sistem ini juga dilengkapi dengan validasi format data dan penyimpanan permanen ke file teks.

Kata Kunci: pengaduan, struktur, queue, stack, hashing, enkripsi.

Abstract

Digital-based public services are increasingly needed to improve efficiency and transparency, one of which is in terms of citizen complaints. This project aims to develop a terminal-based online citizen complaint application using the C++ programming language with the implementation of various data structures. Data structures such as queues are used to manage report queues, stacks for the undo feature, hashing to generate unique report IDs, and encryption to maintain the privacy of reporter data. Sorting is used to organize reports based on priority levels. This system is also equipped with data format validation and permanent storage to text files.

Keywords: complaint, structures, queue, stack, hashing, encryption.

PENDAHULUAN

Pelayanan publik yang cepat, transparan, dan efisien merupakan kebutuhan mendasar dalam masyarakat modern. Salah satu bentuk pelayanan tersebut adalah mekanisme pengaduan dari masyarakat kepada pihak berwenang mengenai berbagai permasalahan yang terjadi di lingkungan sekitar. Di Indonesia, proses pelaporan masalah seperti kerusakan fasilitas umum, gangguan keamanan, atau permasalahan sosial lainnya masih sering dilakukan secara konvensional. Proses ini cenderung memakan waktu, kurang terdokumentasi dengan baik, serta minim umpan balik dari instansi yang menerima laporan. Hal ini menimbulkan kesan bahwa pelaporan masyarakat tidak mendapat perhatian yang semestinya (Notoatmodjo, 2020).

Implementasi sistem pengaduan yang efisien membutuhkan dukungan dari struktur data yang sesuai. Struktur data merupakan fondasi dalam pengembangan perangkat lunak yang berperan penting dalam pengelolaan data. Oleh karena itu, dalam proyek ini digunakan berbagai struktur data seperti *queue*, *sort*, *stack*, enkripsi dan *hashing*. Masing-masing struktur data memiliki karakteristik dan fungsi yang berbeda, yang secara sinergis membentuk sistem pengaduan yang andal (Cormen et al., 2009; Goodrich & Tamassia, 2010).

Queue digunakan untuk mengatur laporan berdasarkan urutan masuk, sementara *sort* memungkinkan laporan yang bersifat mendesak untuk ditangani terlebih dahulu. *Hashing* digunakan untuk menghasilkan ID laporan yang unik serta mencegah duplikasi data (Nayak, 2020). *Stack* dimanfaatkan untuk mencatat riwayat perubahan status laporan. Selain itu, teknik enkripsi diterapkan guna menjaga keamanan dan privasi data pelapor (Jones & George, 2018; Porter & Teisberg, 2006).

Penelitian dan implementasi ini juga mengacu pada studi terdahulu yang menunjukkan efektivitas penggunaan struktur data dalam sistem manajemen, seperti pada sistem rumah sakit dan sistem pelayanan publik lainnya (Dharma & Syahputra, 2017; Hermawan, 2017). Dengan memanfaatkan pendekatan berbasis struktur data, diharapkan sistem yang dibangun mampu menyaring, menyimpan, memproses, dan menampilkan data pengaduan warga secara akurat dan efisien.

Tujuan dari proyek ini adalah untuk mengembangkan sebuah aplikasi pengaduan warga online berbasis teks yang dapat meningkatkan efisiensi dalam penanganan pengaduan, membangun sistem pengaduan yang terorganisir, meningkatkan partisipasi masyarakat dalam pelaporan masalah, serta memberikan solusi bagi instansi dalam pengelolaan pengaduan dengan dukungan struktur data yang tepat.

Metode

Aplikasi pengaduan warga online ini dibuat berbasis terminal menggunakan C++ dan mengintegrasikan struktur data seperti *queue*, *sorting*, *hashing*, *enkripsi*, serta *stack*. Proyek ini menggunakan metode Waterfall karena alur kerjanya terstruktur dan bertahap sesuai dengan kebutuhan pengembangan sistem pengaduan yang jelas dan terdefinisi sejak awal. Berikut implementasi kode dalam sistem:

1. *Sorting* (Prioritas Laporan)

Setelah laporan masuk ke dalam antrian, sistem akan mengevaluasi tingkat urgensinya. Jika laporan termasuk dalam kategori prioritas tinggi, sistem akan menyusun ulang antrian menggunakan *priority queue* atau *heap*. Hal ini memastikan bahwa laporan yang lebih mendesak diproses lebih cepat dibandingkan prioritas lebih rendah.

```
// ===== SORTING (BUBBLE SORT) =====
void sortByPrioritas(MyVector<Pengaduan>& data) {
    for (size_t i = 0; i < data.size(); ++i) {
        for (size_t j = 0; j < data.size() - i - 1; ++j) {
            if (data[j].prioritas > data[j + 1].prioritas) {
                swap(data[j], data[j + 1]);
            }
        }
    }
}
```

Gambar 1 Potongan Kode *Sorting*

2. *Queue* (Antrian Laporan)

Setiap laporan yang dikirim oleh warga akan dimasukkan ke dalam *queue* berdasarkan urutan waktu pengajuan. Sistem memastikan laporan diproses secara sistematis. Laporan yang masuk lebih dulu akan ditangani lebih awal, kecuali jika ada laporan dengan prioritas lebih tinggi.

```
// ===== QUEUE IMPLEMENTATION =====
class QueuePengaduan {
private:
    MyVector<Pengaduan> data;

public:
    void push(const Pengaduan& p) {
        data.push_back(p);
    }

    void pop() {
        if (!empty()) {
            data.erase(0);
        }
    }

    Pengaduan front() const {
        if (!empty()) return data [0];
        throw runtime_error("Queue kosong");
    }

    bool empty() const {
        return data.empty();
    }

    size_t size() const {
        return data.size();
    }

    MyVector<Pengaduan> getAll() const {
        return data;
    }

    void removeByID(size_t id) {
        MyVector<Pengaduan> temp;
        for (size_t i = 0; i < data.size(); ++i) {
            if (data[i].id != id) temp.push_back(data[i]);
        }
        data = temp;
    }
};
```

Gambar 2 Potongan Kode *Queue*

3. *Hashing* (Nomor Laporan)

Setiap laporan yang diajukan akan diberikan nomor laporan unik yang dihasilkan menggunakan teknik hashing. Nomor ini dibuat berdasarkan nama dan isi laporan sebagai identifikasi unik agar admin dapat melacak laporan secara akurat dan mencegah duplikasi.

```
// ===== HASHING ID PENGADUAN =====
size_t generateHashID(const MyString& nama, const MyString& isi) {
    hash<string> hasher;
    return hasher(nama.toStdString() + isi.toStdString());
}
```

Gambar 3 Potongan Kode *Hashing*

4. Enkripsi (Keamanan Data Pelapor)

Untuk menjaga privasi warga, data pelapor seperti nama akan dienkripsi sebelum disimpan. Dengan adanya enkripsi, hanya pihak yang berwenang yang dapat mengakses data pelapor, sehingga keamanan dan kerahasiaan identitas warga tetap terjaga.

```
// ===== ENKRIPSI SEDERHANA =====
MyString enkripsiNama(MyString nama) {
    for (size_t i = 0; i < nama.length(); ++i) {
        nama[i] += 3;
    }
    return nama;
}
```

Gambar 4 Potongan Kode Enkripsi

5. *Stack* (*Undo & Update* Isi Pengaduan)

Setiap perubahan isi pengaduan akan disimpan dalam *stack* sehingga memungkinkan untuk melakukan *undo* jika terjadi kesalahan. Jika pengguna perlu mengembalikan laporan ke pengaduan sebelumnya karena kesalahan, sistem dapat melakukan *rollback*.

```
// ===== STACK IMPLEMENTATION =====
class StackPengaduan {
private:
    MyVector<Pengaduan> data;

public:
    void push(const Pengaduan& p) {
        data.push_back(p);
    }

    Pengaduan pop() {
        if (data.empty()) throw runtime_error("Stack undo kosong");
        Pengaduan topItem = data.back();
        data.pop_back();
        return topItem;
    }

    bool empty() const {
        return data.empty();
    }
};
```

Gambar 5 Potongan Kode Stack

Hasil dan Pembahasan

● Kerumitan Masalah

Tingkat kesulitan proyek sistem pengaduan warga ini bisa dikatakan menengah. Karena sistem pengaduan ini tidak hanya sekadar pencatat data pengaduan warga biasa namun terdapat banyak elemen krusial lainnya yang perlu diperhatikan. Misalnya, ada sistem manajemen antrian berbasis prioritas yang harus bekerja dengan baik, lalu validasi data yang ketat untuk NIK, nomor telepon, dan format tanggal, serta proses penyimpanan data permanen ke file txt. Sistem juga mengimplementasikan enkripsi sederhana untuk menjaga privasi nama warga, selain itu, keberadaan fitur-fitur seperti tambah, edit, hapus, undo, dan finalisasi memerlukan pengelolaan data yang konsisten antar struktur, yakni antara antrian (queue), pemetaan prioritas (map), serta stack untuk undo. Analisis terhadap kerumitan masalah menunjukkan bahwa sistem ini memerlukan pendekatan modular dan terstruktur dalam pengembangannya. Tantangan teknis tidak hanya terletak pada pengelolaan input pengguna yang bervariasi dan berpotensi salah, tetapi juga pada mekanisme pemulihan data (undo) dan pelacakan status pengaduan. Selain itu, karena data harus tersimpan dan dapat dimuat ulang dari file .txt, maka diperlukan desain format file yang stabil serta proses parsing yang aman dan andal.

Tabel 1 Rancangan Program serta Kerumitannya

Aspek	Deskripsi
Jenis Masalah	Sistem informasi berbasis antrian dan pencatatan digital untuk menangani laporan/pengaduan warga secara terstruktur dan terorganisasi.
Sifat Masalah	Kompleksitas sedang. Sistem tidak hanya mencatat data, tetapi juga menangani <i>prioritas</i> , validasi data penting (NIK, telepon, tanggal)
Ruang Lingkup Data	Data pengaduan warga mencakup informasi sensitif dan beragam, seperti nama, NIK, alamat, isi pengaduan, tanggal, dan prioritas, yang membutuhkan validasi dan enkripsi.
Tantangan Teknis	Menjaga konsistensi data antar struktur (queue, map, file).- Enkripsi/dekripsi sederhana untuk menjaga privasi.- Sinkronisasi antar struktur data.

Aspek Keamanan & Integritas	Penggunaan enkripsi nama dan penguncian (<i>finalisasi</i>) data setelah diselesaikan menunjukkan adanya kebutuhan akan integritas dan keamanan data warga.
Interaksi dan Operasi	Sistem mendukung berbagai operasi seperti tambah, edit, hapus, tampilkan, undo, finalisasi—yang berpotensi menimbulkan inkonsistensi jika tidak dikelola baik.
Persistensi Data	Data harus disimpan secara permanen di file .txt, dan harus dapat dimuat kembali secara utuh dan terstruktur saat aplikasi dijalankan ulang.
Kebutuhan Real-time	Meski tidak berbasis jaringan, sistem harus responsif dan memperbarui antrian serta status pengaduan secara langsung saat dilakukan operasi (edit, undo, delete).

● **Tingkat kerumitan kode program dan struktur data yang dibuat**

1. Struktur Data Pengaduan

Menyimpan data pengaduan sebagai satu unit terstruktur. Kompleksitas rendah karena hanya berfungsi sebagai penampung data, tanpa operasi algoritmik kompleks di dalamnya.

2. Kelas Queue Pengaduan

Mengatur antrian pengaduan menggunakan STL queue, dengan operasi cepat $O(1)$. Namun, pencarian atau pengurutan data membutuhkan iterasi tambahan yang menambah kompleksitas menjadi $O(n)$.

3. Kelas Stack Pengaduan

Menyimpan riwayat perubahan pengaduan untuk fitur undo. Operasi $O(1)$, namun penggunaannya memperkaya logika aplikasi dan memungkinkan pemulihan data secara efisien dan terstruktur.

4. Fungsi Validasi

Memastikan input sesuai format untuk NIK, telepon, dan tanggal. Operasi berbasis string dan parsing, kompleksitas rendah namun penting untuk menjaga integritas data pengguna.

5. Fungsi enkripsi Nama

Melakukan enkripsi sederhana terhadap nama dengan Caesar cipher. Kompleksitas rendah $O(k)$, tetapi cukup efektif untuk melindungi privasi pengguna dalam penyimpanan file.

6. Fungsi simpan ke File

Menulis seluruh data ke file CSV. Membutuhkan iterasi penuh atas seluruh map, menjadikan kompleksitasnya $O(n)$ setiap kali terjadi penyimpanan data secara total.

7. Fungsi tambah Pengaduan

Menggabungkan validasi, enkripsi, dan penyimpanan data baru. Kompleksitas tiap bagian relatif rendah, namun secara keseluruhan fungsi ini penting sebagai pintu masuk utama data.

8. Fungsi tampilkan Pengaduan

Menampilkan semua pengaduan dan mengurutkannya berdasarkan prioritas. Bubble sort menyebabkan kompleksitas tinggi $O(n^2)$, sehingga berpotensi lambat pada data besar.

9. Fungsi ubah Pengaduan

Memungkinkan pengguna mengedit pengaduan selama belum final. Kompleksitas rendah $O(n)$, tetapi terintegrasi dengan stack untuk undo, meningkatkan fleksibilitas sistem.

10. Fungsi hapus Pengaduan

Menghapus entri dari map dan file, penulisan ulang file membuat proses berskala $O(n)$ dan perlu efisiensi lebih lanjut.

11. Fungsi selesaiPengaduan

Menandai pengaduan sebagai selesai, mencegah perubahan lanjutan. Memiliki kompleksitas rendah, tetapi menambah logika status yang penting untuk kontrol data.

12. Fungsi undo

Mengembalikan kondisi pengaduan sebelum diubah atau diselesaikan. Menggunakan stack, operasinya cepat, namun tidak bisa mengubah status final secara langsung.

13. Fungsi menu

Mengatur navigasi utama aplikasi. Kompleksitas sangat rendah karena hanya mengarahkan pengguna ke fungsi sesuai pilihan, tanpa proses algoritmik berat.

- **Jenis operasi struktur data yang digunakan**

Aplikasi pengaduan warga menggunakan beberapa struktur data dasar yang digunakan untuk menyimpan, memproses, dan memanipulasi data pengaduan. Struktur data pertama yang digunakan adalah struct Pengaduan, yang merepresentasikan satu entitas pengaduan, mencakup atribut seperti nama, nik, alamat, isiPengaduan, status, tanggal, dan prioritas. Ini adalah representasi dasar dari unit data.

Untuk menyimpan dan mengatur urutan pengaduan secara dinamis, digunakan MyVector, yang berperan sebagai array dinamis. MyVector digunakan di berbagai bagian aplikasi, seperti pada kelas QueuePengaduan dan StackPengaduan, yang masing-masing mengimplementasikan struktur data Queue (antrian) dan Stack (tumpukan). Queue digunakan untuk memproses pengaduan sesuai urutan masuk, sedangkan Stack dimanfaatkan untuk fitur undo, memungkinkan pengguna membatalkan perubahan terakhir.

Selain itu, kode ini menggunakan unordered_map, yaitu struktur data hash table, dalam dua konteks: pertama, laporanMap menyimpan pengaduan berdasarkan ID sebagai kunci, memungkinkan pencarian cepat, kedua finalPengaduan menyimpan status pengaduan (apakah sudah final atau belum) juga berbasis ID. Fungsi hashing juga dimanfaatkan untuk membuat ID unik dengan generateHashID(), menggunakan gabungan nama dan isi pengaduan.

Dalam hal algoritma, kode ini juga menerapkan bubble sort untuk mengurutkan pengaduan berdasarkan prioritas dalam fungsi sortByPrioritas. Meskipun tidak efisien untuk skala besar, bubble sort mudah diimplementasikan dan cukup untuk jumlah data terbatas.

Program ini juga melibatkan operasi file I/O untuk menyimpan dan memuat data pengaduan dari dan ke file teks (data_pengaduan.txt), serta menggunakan validasi string untuk memeriksa format tanggal, NIK, dan nomor telepon. Nama pelapor dienkripsi dengan operasi manipulasi karakter (menambahkan 3 ke setiap karakter).

- **Hasil dari produk yang dibuat**

Program ini merupakan aplikasi sistem untuk pengaduan warga yang berbasis terminal, ditulis dalam bahasa C++. Tujuan dari aplikasi ini adalah untuk mendata, mengelola, dan menyimpan laporan pengaduan dari masyarakat secara sistematis dan aman. Produk ini memiliki fitur yang cukup lengkap, meliputi penambahan laporan pengaduan, menampilkan laporan yang ada, menyelesaikan pengaduan, mengedit isi laporan, menghapus laporan, dan bahkan mendukung fitur undo untuk membatalkan

perubahan terakhir. Semua data disimpan di dalam file `data_pengaduan.txt`. Dari sisi keamanan dan integritas data, program ini menerapkan enkripsi sederhana pada nama pelapor. Nama asli tetap disimpan secara terpisah dalam file txt, dan di terminal pengguna tetap dapat menampilkan hasil pengaduan dengan keluaran masih nama asli. Selain itu, setiap laporan diberikan ID unik yang dihasilkan menggunakan fungsi hash berbasis nama dan isi laporan, memastikan tidak ada ID yang duplikat.

- **Interface**

Interface dari program ini adalah antarmuka teks (Command-line Interface/CLI). CLI adalah antarmuka pengguna (UI) berbasis teks yang digunakan untuk berinteraksi dengan komputer (Rosnelly & Yudha, 2023). Aplikasi ini merupakan sistem pengaduan warga yang memungkinkan pengguna untuk menambahkan, menampilkan, mengedit, menyelesaikan, menghapus, dan meng-undo pengaduan dengan berbagai validasi input dan penyimpanan ke *file*.

```
=== Aplikasi Pengaduan Warga ===
1. Tambah Pengaduan
2. Tampilkan Pengaduan
3. Selesaikan Pengaduan
4. Edit Isi Pengaduan
5. Undo Perubahan
6. Hapus Pengaduan
7. Keluar
Pilih menu: █
```

Gambar 6 Tampilan Menu Utama

Pengguna akan diminta memasukkan sejumlah informasi yang dibutuhkan. Setiap masukan yang diberikan oleh pengguna akan divalidasi agar sesuai dengan format dan kriteria yang telah ditentukan. Jika terdapat kesalahan dalam format pengisian data, program akan memberikan pesan kesalahan validasi dan meminta pengguna untuk mengulang input hingga benar. Semua interaksi ini memastikan pengguna dapat memantau status pengaduan secara transparan dan akurat.

- **Alur Program Aplikasi**

1. **Tambah Pengaduan**

```
Pilih menu: 1
Masukkan nama pelapor: Wahyu
Masukkan tanggal pengaduan (dd/mm/yy): 28/05/25
Masukkan NIK (16 digit): 0000000000000000
Masukkan nomor telepon (10-13 digit): 123123123123
Masukkan alamat: Jalan Sukahati
Masukkan isi pengaduan: Pepohonan runtuh akibat badai dahsyat
Masukkan prioritas (1-5): 3
Pengaduan berhasil ditambahkan! ID : 1103408368826710147
```

Gambar 7 Tampilan Menu Tambah Pengaduan

Pengguna menambahkan laporan pengaduan baru dengan mengisi data seperti nama, tanggal, NIK, nomor telepon, alamat, isi pengaduan, dan tingkat prioritas. Sistem akan memvalidasi input NIK dan nomor telepon. Selanjutnya, data disimpan ke dalam *queue* dan dicatat dalam *map*. Setelah selesai, sistem menampilkan ID pengaduan.

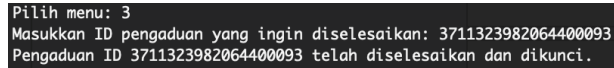
2. **Tampilkan Pengaduan**

```
Pilih menu: 2
Tanggal: 28/05/25
ID: 1103408368826710147
Nama: Wahyu
NIK: 0000000000000000
Alamat: Jalan Sukahati
Telepon: 123123123123
Isi Pengaduan: Pepohonan runtuh akibat badai dahsyat
Status: Terselesaikan
Prioritas: 3
-----
```

Gambar 8 Tampilan Menu Tampilkan Pengaduan

Menu ini untuk menampilkan seluruh pengaduan yang telah ditambahkan ke dalam sistem. Data pengaduan akan diambil dari *queue* dan diurutkan secara *ascending* berdasarkan tingkat prioritas menggunakan algoritma *bubble sort*, dari prioritas tertinggi (angka terkecil) ke terendah. Tujuannya adalah agar petugas atau admin dapat melihat pengaduan dan segera memproses laporan dengan prioritas lebih tinggi.

3. Selesaikan Pengaduan

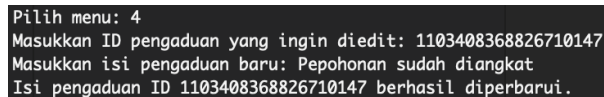


```
Pilih menu: 3
Masukkan ID pengaduan yang ingin diselesaikan: 3711323982064400093
Pengaduan ID 3711323982064400093 telah diselesaikan dan dikunci.
```

Gambar 9 Tampilan Menu Selesaikan Pengaduan

Menu ini dapat menandai sebuah pengaduan sebagai selesai dan final. Sistem akan mengecek ID tersebut valid dan belum bersifat final. Jika valid, maka status pengaduan akan diperbarui menjadi "Terselesaikan". Pengaduan tidak dapat diubah lagi sehingga menjaga integritas laporan yang sudah diselesaikan.

4. Edit isi pengaduan

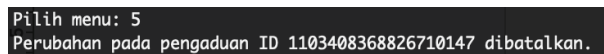


```
Pilih menu: 4
Masukkan ID pengaduan yang ingin diedit: 1103408368826710147
Masukkan isi pengaduan baru: Pepohonan sudah diangkat
Isi pengaduan ID 1103408368826710147 berhasil diperbarui.
```

Gambar 10 Tampilan Menu Edit Isi Pengaduan

Menu ini untuk mengubah isi laporan pengaduan yang telah dimasukkan sebelumnya, selama pengaduan tersebut belum memiliki status selesaikan pengaduan. Sistem akan meminta ID pengaduan yang akan diedit. Setelah itu, sistem akan memeriksa validitas ID dan memastikan pengaduan belum bersifat final. Jika valid, maka isi pengaduan diperbarui. Setelah perubahan data akan disinkronkan dan *file read* akan diperbarui.

5. Undo Perubahan

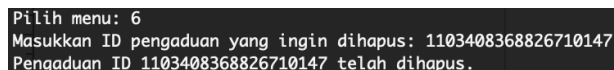


```
Pilih menu: 5
Perubahan pada pengaduan ID 1103408368826710147 dibatalkan.
```

Gambar 11 Tampilan Menu Undo Perubahan

Menu ini untuk membatalkan perubahan terakhir pembaruan pada pengaduan yang belum final. Sistem akan mengambil data terakhir dari stack undo dan mengembalikannya ke kondisi sebelum perubahan. Data dalam *queue* dan *map* akan diperbarui sesuai dengan isi pengaduan sebelumnya. Jika pengaduan sudah dalam status final, maka proses *undo* tidak dapat dilakukan.

6. Hapus Pengaduan



```
Pilih menu: 6
Masukkan ID pengaduan yang ingin dihapus: 1103408368826710147
Pengaduan ID 1103408368826710147 telah dihapus.
```

Gambar 12 Tampilan Menu Hapus Pengaduan

Menu ini dapat menghapus pengaduan dengan meminta ID pengaduan yang ingin dihapus. Jika pengaduan ditemukan dan belum final, maka sistem akan menghapus data tersebut dari *queue*, *map*, serta status final. Jika memilih tampilkan pengaduan maka outputnya adalah *Tidak ada pengaduan saat ini*. Jika pengaduan sudah final, maka sistem akan menolak permintaan penghapusan dengan memberikan output *Pengaduan ini sudah final dan tidak dapat dihapus*.

7. Keluar



```
Pilih menu: 7
Terima kasih telah menggunakan aplikasi ini!
```

Gambar 13 Tampilan Menu Keluar

Menu ini akan mengakhiri program dan menutup sesi penggunaan sistem pengaduan. Selama sesi berjalan, seluruh perubahan telah disimpan secara otomatis ke dalam *file read*. Oleh karena itu, keluar dari aplikasi tidak akan menyebabkan kehilangan data.

- **Analisis Kompleksitas**

Dalam pengembangan aplikasi, setiap fitur utama dianalisis berdasarkan kompleksitas waktunya. Kompleksitas waktu adalah jumlah operasi yang dilakukan untuk melaksanakan algoritma sebagai fungsi dari ukuran masukan n (Rahayuningsih, 2016). Berikut adalah kompleksitas untuk setiap fungsi utamanya:

- a. Menambahkan pengaduan memiliki kompleksitas $O(1)$ karena proses ini hanya melakukan penambahan data dalam waktu konstan. Data juga ditulis tanpa perulangan sehingga tidak mengganggu kompleksitas.
- b. Menampilkan pengaduan memiliki kompleksitas $O(n^2)$ karena data disalin dari *queue* menuju *array* kemudian dilakukan *bubble sort*.
- c. Mengedit isi pengaduan memiliki kompleksitas $O(n)$ karena program mencari dan mengganti data dalam *queue* serta memperbaruinya.
- d. Menyelesaikan pengaduan memiliki kompleksitas $O(n)$ karena program mencari dan menyimpan dalam *stack*.
- e. Meng-undo pengaduan memiliki kompleksitas $O(n)$ karena data dari *stack* dikembalikan kembali lalu dimasukkan ke *queue* dan ditulis ulang seluruhnya.
- f. Menghapus pengaduan memiliki kompleksitas $O(n)$ karena penghapusan data di *queue* dilakukan secara linier.

- **Efisiensi**

Aplikasi pengaduan warga online ini cukup efisien untuk lingkup warga kecil hingga menengah. Penggunaan struktur data dan memori yang ringan dalam aplikasi mendukung kinerja pengelolaan data dengan cepat. Penyimpanan dataset menuju file write/read dapat memastikan data tetap tersimpan aman. Namun penggunaan algoritma bubble sort masih kurang efisien sehingga masih dapat dikembangkan kembali menjadi aplikasi yang seutuhnya efisien.

- **Kelebihan dan Kekurangan**

Kelebihan:

1. Validasi input: Data dipastikan akurat karena terjadi verifikasi tanggal, NIK, dan nomor telepon.
2. Adanya fitur edit dan undo: Pengguna mendapatkan pengalaman fleksibel untuk melakukan perbaikan data yang telah dikirim sebelum data di finalisasi.
3. Penyimpanan di file lokal: Data yang telah masuk disimpan dalam file lokal sehingga memudahkan untuk pengarsipan data.

Kekurangan:

1. Skala yang kecil: Aplikasi ini belum optimal untuk skala besar karena penyimpanan yang digunakan masih sederhana.
2. *Undo* terbatas: Fitur undo yang menggunakan stack hanya menyimpan tindakan terakhir dan hanya meng-undo perubahan isi pengaduan.
3. Pencarian hanya berdasarkan ID tanpa fitur yang lain seperti tanggal atau status pengaduan.

Simpulan

Sistem pengaduan warga online yang telah dikembangkan memanfaatkan berbagai struktur data seperti queue, stack, hashing, dan map untuk mendukung proses manajemen laporan secara efisien. Meskipun algoritma bubble sort digunakan untuk penyusunan prioritas laporan, metode ini masih dapat dioptimalkan lebih lanjut dengan algoritma sorting yang lebih efisien seperti quicksort. Sistem ini mampu menangani proses pengaduan secara berurutan berdasarkan prioritas, identifikasi laporan secara unik, serta menjaga keamanan dan integritas data warga melalui validasi dan enkripsi

Aplikasi ini harapannya mendapat pengembangan seperti menambahkan sistem pembeda antara user dengan admin yang memungkinkan kontrol akses terhadap fitur-fitur tertentu. Selain itu adanya log untuk mencatat riwayat perubahan pengaduan sebagai transparansi data.

Ucapan Terima Kasih

Puji dan syukur kami panjatkan ke hadirat Allah SWT atas segala limpahan rahmat, karunia, dan kemudahan-Nya sehingga kami dapat menyelesaikan laporan ini dengan baik. Kami menyadari bahwa pencapaian ini tidak lepas dari bantuan, dukungan, dan kerja sama dari berbagai pihak sepanjang proses penyusunan proyek ini.

Kami mengucapkan terima kasih yang sebesar-besarnya kepada dosen mata kuliah yang telah memberikan ilmu, dan arahan yang sangat berarti selama proses perkuliahan dan pengerjaan proyek ini. Ucapan terima kasih juga kami sampaikan kepada asisten praktikum P2, Kak Meena dan Kak Alya, atas bimbingan dan bantuan teknis yang sangat membantu dalam menyelesaikan bagian dari proyek ini.

Tak lupa, kami juga mengucapkan terima kasih kepada teman-teman kelompok yang telah bekerja sama dengan penuh semangat dan tanggung jawab hingga proyek ini dapat diselesaikan dengan baik. Semoga segala kebaikan dan bantuan yang telah diberikan menjadi amal yang bernilai di sisi-Nya.

Daftar Pustaka

- Cormen TH, Leiserson CE, Rivest RL, Stein C. 2009. *Introduction to Algorithms*. Cambridge (US): MIT Press.
- Dharma A, Syahputra HHP. 2017. Aplikasi Pembelajaran Linked List Berbasis Mobile Learning. *Riau J Comput Sci*. 4(1):1–11.
- Goodrich MT, Tamassia R. 2010. *Data Structures and Algorithms in C++*. Hoboken (US): John Wiley & Sons.
- Hermawan W. 2017. *Manajemen Pelayanan Kesehatan*. Bandung (ID): Alfabeta.
- Jones GR, George JM. 2018. *Essentials of Contemporary Management*. New York (US): McGraw-Hill Education.
- Notoatmodjo S. 2020. *Kesehatan Masyarakat: Prinsip-Prinsip dan Aplikasinya*. Jakarta (ID): Rineka Cipta.
- Porter ME, Teisberg EO. 2006. *Redefining Health Care: Creating Value-Based Competition on Results*. Boston (US): Harvard Business Review Press.
- Rahayuningsih P A. 2016. Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). *J Evolusi*. 4(2): 64-75.
- Rosnelly R, Yudha S W. 2023. Perancangan Aplikasi Otomatisasi Deteksi XSS Berbasis CLI dengan Bahasa Pemrograman GO. *J Information System*. 8(2): 11-20.