

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data STI

PURRMART

Dipersiapkan oleh:

Kelompok K2-07

Ahmad Evander Ruizhi Xavier / 18223064

Nazwan Siddqi Muttaqin / 18223066


Keane Putra Lim / 18223056

Sebastian Albern Nugroho / 18223074

Joan Melkior Silaen / 18223102

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB-02-07</i>		<i>58</i>
		<i>Revisi</i>	<i><no revisi></i>	<i>19-12-2024</i>

Daftar Isi

1	Ringkasan.....	4
2	Penjelasan Tambahan Spesifikasi Tugas.....	5
2.1	Spesifikasi Fitur Save.....	5
2.2	Spesifikasi Fitur Swap <i> <j>.....	5
2.3	Spesifikasi Fitur Riwayat Maksimal.....	5
2.4	Spesifikasi Fitur Optimasi Rute Ekspedisi.....	5
2.5	Spesifikasi Fitur Profile.....	5
3	Struktur Data (ADT).....	6
3.1	Stack.....	6
3.2	ListLinier.....	7
3.3	Map.....	10
4	Program Utama.....	11
5	Algoritma-Algoritma Menarik.....	30
5.1	Algoritma Random Number Generator.....	30
5.2	Algoritma Delay Waktu.....	30
5.3	Algoritma Pewarnaan Teks Terminal.....	31
6	Data Test.....	32
6.1	PROFILE.....	32
6.2	CART ADD <nama> <n>.....	32
6.3	CART REMOVE <nama> <n>.....	33
6.4	CART SHOW.....	34
6.5	CART PAY.....	35
6.6	HISTORY <n> (Bonus).....	37
6.7	WISHLIST ADD.....	38
6.8	WISHLIST SWAP <i> <j>.....	39
6.9	WISHLIST REMOVE <i>.....	40
6.10	WISHLIST REMOVE.....	41
6.11	WISHLIST CLEAR.....	42
6.12	WISHLIST SHOW.....	42
7	Test Script.....	43
8	Pembagian Kerja dalam Kelompok.....	48
9	Lampiran.....	49
9.1	Deskripsi Tugas Besar.....	49
9.2	Notulen Rapat.....	53
9.3	Log Activity Anggota Kelompok.....	57

1 Ringkasan

PURRMART adalah aplikasi simulasi e-commerce berbasis CLI (command-line interface) yang dikembangkan sebagai tugas besar mata kuliah Algoritma dan Struktur Data STI. Aplikasi ini dirancang dalam konteks misi rahasia OWCA (Organization Without a Cool Acronym) untuk menghentikan rencana jahat Dr. Asep Spakbor yang ingin menghancurkan tiga wilayah negara bagian dengan mesin "Oppenheimer-inator".

Fitur utama PURRMART meliputi:

1. Menampilkan dan mengelola barang di toko
2. Meminta dan menyuplai barang baru
3. Menyimpan dan membeli barang dalam keranjang
4. Menampilkan riwayat pembelian
5. Membuat dan menghapus wishlist
6. Sistem kerja untuk menghasilkan uang

Aplikasi ini diimplementasikan menggunakan bahasa C dengan memanfaatkan berbagai struktur data yang telah dipelajari, seperti array dinamis, linked list, queue, dan mesin kata. Tantangan utama dalam pengembangan PURRMART adalah merancang basis data relasional yang efisien untuk mengelola dependensi fungsional, terutama dalam proses restock barang, sinkronisasi wishlist dan keranjang, serta pelacakan riwayat pembelian.

Pengembangan PURRMART juga mencakup fitur-fitur tambahan seperti sistem login/register, penyimpanan dan pemuatan konfigurasi, serta mini-game untuk menghasilkan uang tambahan. Implementasi dilakukan dengan memperhatikan efisiensi penggunaan memori dan kecepatan eksekusi, mengingat batasan penggunaan library yang hanya meliputi stdio.h, stdlib.h, time.h, dan math.h.

Hasil dari tugas besar ini mendemonstrasikan pemahaman dan penerapan konsep-konsep algoritma dan struktur data dalam pengembangan aplikasi yang kompleks dan interaktif. PURRMART tidak hanya berfungsi sebagai simulasi e-commerce, tetapi juga sebagai alat

pembelajaran yang efektif untuk mahasiswa dalam mengaplikasikan teori ke dalam praktik pemrograman.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Spesifikasi Fitur Save

Fitur save yang digunakan akan mengecek apakah input merupakan file konfigurasi (config.txt) atau input bukan dalam file txt. Jika iya, maka save akan gagal dan user diminta untuk mengulangi input hingga benar.

2.2 Spesifikasi Fitur Swap <i> <j>

Fitur swap yang digunakan juga akan mengecek jika kedua posisi masukkan tidak terdefinisi dan masukkan yang tidak valid.

2.3 Spesifikasi Fitur Riwayat Maksimal

Fitur riwayat maksimal membuat riwayat pembelian dari setiap pengguna disimpan secara detail dengan menunjukkan seluruh barang yang dibeli.

2.4 Spesifikasi Fitur Optimasi Rute Ekspedisi

Fitur optimasi rute ekspedisi ini memungkinkan ekspedisi untuk mengirim barang dengan rute yang paling efisien. Pada fitur ini kami menggunakan algoritma gabungan dari permutasi dan brute force. Kami memilih algoritma ini karena cukup sederhana untuk diimplementasikan lalu dengan mencoba semua kemungkinan permutasi rute yang valid, kita dijamin menemukan rute terpendek yang mengunjungi semua titik.

2.5 Spesifikasi Fitur Profile

Fitur tambahan pada profile ini memungkinkan pengguna untuk melihat total pengeluaran dia, total penghasilan dia, dan juga banyaknya barang yang dia punya berdasarkan riwayat pembeliannya.

3 Struktur Data (ADT)

3.1 Stack

1. CreateEmptyStack(Stack S)

Fungsi ini digunakan untuk membuat sebuah stack baru yang kosong dengan kapasitas maksimum $MaxEl$. Indeks TOP diatur ke Nil, menandakan bahwa stack belum memiliki elemen.

2. IsEmptyStack(Stack S)

Fungsi ini memeriksa apakah stack S kosong dengan memeriksa apakah nilai TOP adalah Nil. Mengembalikan true jika stack kosong, dan false jika tidak.

3. IsFullStack(Stack S)

Fungsi ini memeriksa apakah stack S sudah penuh dengan memeriksa apakah nilai TOP sama dengan $MaxEl - 1$. Mengembalikan true jika stack penuh, dan false jika tidak.

4. Push(Stack S, infotypeStack X)

Fungsi ini menambahkan elemen X ke dalam stack S. Jika stack tidak penuh, nilai TOP akan bertambah satu, dan X akan menjadi elemen baru di posisi TOP.

5. Pop(Stack S, infotypeStack X)

Fungsi ini menghapus elemen dari stack S. Elemen yang dihapus adalah elemen di posisi TOP, dan nilai tersebut disimpan dalam variabel X. Setelah penghapusan, nilai TOP akan berkurang satu.

6. CopyStack(Stack S, Stack copy)

Fungsi ini menyalin semua elemen dari stack S ke stack copy. Pertama, elemen-elemen dari S dihapus dan disimpan dalam stack sementara, kemudian elemen-elemen tersebut dipindahkan kembali ke copy dan S untuk menjaga urutan elemen.

3.2 ListLinier

1. IsEmptyLink(LinkedList L)

Fungsi ini memeriksa apakah list L kosong dengan memeriksa apakah elemen pertama (First) bernilai Nil. Mengembalikan true jika list kosong, dan false jika tidak.

2. CreateEmptyLink(LinkedList L)

Fungsi ini digunakan untuk membuat list baru yang kosong. Indeks pertama dari list diatur ke Nil, menandakan bahwa list belum memiliki elemen.

3. Alokasi(infotype X)

Fungsi ini mengalokasikan memori untuk elemen baru dengan nilai X. Jika alokasi berhasil, fungsi ini mengembalikan alamat elemen yang baru dialokasikan; jika gagal, mengembalikan Nil.

4. Dealokasi(address P)

Fungsi ini mengembalikan alamat P ke sistem, melakukan dealokasi memori yang sebelumnya dialokasikan untuk elemen list.

5. SearchLink(LinkedList L, infotype X)

Fungsi ini mencari elemen dalam list L yang memiliki nilai X. Jika elemen ditemukan, fungsi ini mengembalikan alamat elemen tersebut; jika tidak, mengembalikan Nil.

6. InsVFirst(LinkedList L, infotype X)

Fungsi ini menambahkan elemen baru dengan nilai X di awal list L. Jika alokasi berhasil, elemen baru akan menjadi elemen pertama.

7. InsVLast(LinkedList L, infotype X)

Fungsi ini menambahkan elemen baru dengan nilai X di akhir list L. Jika alokasi berhasil, elemen baru akan menjadi elemen terakhir.

8. DelVFirst(LinkedList L, infotype X)

Fungsi ini menghapus elemen pertama dari list L. Nilai elemen yang dihapus disimpan dalam X, dan alamat elemen tersebut akan di-dealokasi.

9. DelVLast(LinkedList L, infotype X)

Fungsi ini menghapus elemen terakhir dari list L. Nilai elemen yang dihapus disimpan dalam X, dan alamat elemen tersebut akan di-dealokasi.

10. InsertFirstLink(LinkedList L, address P)

Fungsi ini menambahkan elemen dengan alamat P sebagai elemen pertama dalam list L. Elemen baru akan menunjuk ke elemen pertama yang lama.

11. InsertAfter(LinkedList L, address P, address Prec)

Fungsi ini menyisipkan elemen dengan alamat P setelah elemen yang ditunjuk oleh Prec. Elemen baru akan menunjuk ke elemen yang sebelumnya berada setelah Prec.

12. InsertLastLink(LinkedList L, address P)

Fungsi ini menambahkan elemen dengan alamat P sebagai elemen terakhir dalam list L. Jika list kosong, elemen baru akan menjadi elemen pertama.

13. DelFirst(LinkedList L, address P)

Fungsi ini menghapus elemen pertama dari list L dan menyimpan alamat elemen yang dihapus dalam P. Elemen baru yang menjadi pertama akan ditunjuk oleh First.

14. DelP(LinkedList L, infotype X)

Fungsi ini menghapus elemen dari list L yang memiliki nilai X. Jika elemen ditemukan, elemen tersebut akan di-dealokasi; jika tidak, list tetap tidak berubah.

15. DelLast(LinkedList L, address P)

Fungsi ini menghapus elemen terakhir dari list L dan menyimpan alamat elemen yang dihapus dalam P. Jika list menjadi kosong setelah penghapusan, First akan diatur ke Nil.

16. DelAfter(LinkedList L, address Pdel, address Prec)

Fungsi ini menghapus elemen yang berada setelah elemen yang ditunjuk oleh Prec. Alamat elemen yang dihapus disimpan dalam Pdel.

17. PrintInfo(LinkedList L)

Fungsi ini mencetak semua elemen dalam list L dalam format yang terstruktur, yaitu di antara tanda kurung siku. Jika list kosong, hanya menampilkan tanda kurung siku kosong.

18. NbElmt(LinkedList L)

Fungsi ini menghitung dan mengembalikan jumlah elemen dalam list L. Jika list kosong, mengembalikan 0.

3.3 Map

1. CreateEmptyMap(Map M)

Fungsi ini digunakan untuk membuat sebuah Map baru yang kosong. Setelah pemanggilan fungsi ini, jumlah elemen (Count) dalam Map diatur ke NilM, menandakan bahwa Map belum memiliki elemen.

2. IsEmptyMap(Map M)

Fungsi ini memeriksa apakah Map M kosong dengan memeriksa apakah Count sama dengan NilM. Mengembalikan true jika Map kosong, dan false jika tidak.

3. IsFull(Map M)

Fungsi ini memeriksa apakah Map M sudah penuh dengan memeriksa apakah Count sama dengan MaxEl. Mengembalikan true jika Map penuh, dan false jika tidak.

4. Value(Map M, keytype k)

Fungsi ini mengembalikan nilai (valuetype) yang berpasangan dengan kunci k dalam Map M. Jika k tidak ditemukan, fungsi ini mengembalikan Undefined.

5. InsertMap(Map M, keytype k, valuetype v)

Fungsi ini menambahkan elemen dengan kunci k dan nilai v ke dalam Map M. Jika k sudah ada dalam Map, fungsi ini tidak melakukan apa-apa.

6. DeleteMap(Map M, keytype k)

Fungsi ini menghapus elemen dengan kunci k dari Map M. Jika k ditemukan, semua elemen setelahnya akan digeser ke kiri untuk mengisi celah yang ditinggalkan, dan Count akan berkurang.

7. IsMember(Map M, keytype k)

Fungsi ini memeriksa apakah k adalah anggota dari Map M. Mengembalikan true jika k ditemukan, dan false jika tidak.

4 Program Utama

Pada file main(), program akan menghasilkan >>> agar user mampu input perintah. Input bisa berupa start, load. Lalu bisa dilanjutkan dengan login, register untuk daftar sebagai user.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "console.h"

int main() {
    int nBarang = 0;
    int nUser = 0;
    int userIdx = -1;
    ArrayDin barang = MakeArrayDin();
    List user = MakeList();
    Queue antrian;
    CreateQueue(&antrian);

    boolean loaded = false; // untuk start dan load
    boolean loggedIn = false; // untuk login
    boolean change = false; // untuk save

    while(true){
        while (!loaded){
```

```

system("cls || clear");
displayPurrMart();
struct timespec req = {1, 0};
nanosleep(&req, NULL);
displayWelcomeMenu();
printf(COLOR_BOLD_BLUE"Masukkan perintah Anda: "COLOR_OFF);
STARTWORD();
Word choice = CurrentWord;

while (currentChar != '\n'){
ADVWORD();
choice.TabWord[choice.Length] = ' ';
choice.Length++;

    for (int i = 0; i < CurrentWord.Length; i++){
        choice.TabWord[choice.Length] = CurrentWord.TabWord[i];
        choice.Length++;
    }
}

if (isKataSama(choice, "START")) {
    start(&barang, &user, &nBarang, &nUser);

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
    loaded = true;
}
else if (startsWith(choice, "LOAD")) {
    char loadfile[100] = "save/";

```

```

        int loadfileLen = 5;
        int i = 5;

        // Skip any spaces after LOAD
        while (i < choice.Length && choice.TabWord[i] == ' ') {
            i++;
        }

        while (i < choice.Length) {
            loadfile[loadfileLen++] = choice.TabWord[i];
            i++;
        }
        loadfile[loadfileLen] = '\\0';

        printf(COLOR_BOLD_CYAN"\\n>> LOAD %s\\033[0m\\n", loadfile +
5);

        load(loadfile, &barang, &user, &nBarang, &nUser);

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
        if (nUser > 0){
            loaded = true;
        }
    }
    else if (isKataSama(choice, "QUIT")) {
        printf(COLOR_BOLD_CYAN"\\n>> QUIT\\n\\n"COLOR_OFF);
        quit(barang, user, nBarang, nUser, change);
    }
    else if (isKataSama(choice, "HELP")) {

```

```

        printf(COLOR_BOLD_CYAN"\n>> HELP\n\n"COLOR_OFF);
        help1();

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }
    else {
        printf("Menu tidak valid, silahkan coba lagi\n");

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
}

while(!loggedIn){
    system("cls || clear");
    displayPurrMart();
    displayLoginMenu();
    printf(COLOR_BOLD_BLUE"Masukkan perintah Anda: "COLOR_OFF);
    STARTWORD();
    Word choice = CurrentWord;

    if (isKataSama(choice, "REGISTER")) {
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\n>> REGISTER\n\n"COLOR_OFF);
        addUser(&user, &nUser);
        change = true;
    }
}

```

```

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "LOGIN")) {
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\n>> LOGIN\n\n"COLOR_OFF);
        userIdx = loginUser(user, nUser);

        if (userIdx != -1) {
            struct timespec req = {2, 0};
            nanosleep(&req, NULL);
            loggedIn = true;
            break;
        }

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "QUIT")) {
        if (change){
            system("cls || clear");
            displayPurrMart();
            printf(COLOR_BOLD_CYAN"\n>> QUIT\n\n"COLOR_OFF);
            quit(barang, user, nBarang, nUser, change);
        }
        else {
            printf(COLOR_BOLD_CYAN"\n>> QUIT\n\n"COLOR_OFF);
            quit(barang, user, nBarang, nUser, change);
        }
    }

```

```

        }
    }
    else if (isKataSama(choice, "HELP")) {
        printf(COLOR_BOLD_CYAN"\n>> HELP\n\n"COLOR_OFF);
        help2();

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }
    else {
        printf("Menu tidak valid, silahkan coba lagi\n");

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
}

while(loggedIn){
    system("cls || clear");
    displayPurrMart();
    displayMainMenu();
    printf(COLOR_BOLD_BLUE"Masukkan perintah Anda: "COLOR_OFF);
    STARTWORD();
    Word choice = CurrentWord;

    while (currentChar != '\n'){
        ADVWORD();
        choice.TabWord[choice.Length] = ' ';
        choice.Length++;
    }
}

```

```

        for (int i = 0; i < CurrentWord.Length; i++){
            choice.TabWord[choice.Length] = CurrentWord.TabWord[i];
            choice.Length++;
        }
    }

    if (isKataSama(choice, "PROFILE")) {
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\n>> PROFILE\033[0m\n\n");
        profile(user, userIdx);
        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }

    else if (isKataSama(choice, "WORK")) {
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\n>> WORK\033[0m\n\n");
        work(&user, userIdx);
        change = true;

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }

    else if (isKataSama(choice, "WORK CHALLENGE")) {
        system("cls || clear");
        displayPurrMart();
    }

```



```

        printf(COLOR_BOLD_CYAN"\n>> WORK CHALLENGE\033[0m\n\n");
        work_challenge(&user, userIdx);
        change = true;

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "STORE LIST")) {
        printf(COLOR_BOLD_CYAN"\n>> STORE LIST\033[0m\n\n");
        storeList(barang, nBarang);

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "STORE REQUEST")) {
        printf(COLOR_BOLD_CYAN"\n>> STORE REQUEST\033[0m\n\n");
        storeRequest(&barang, &antrian);

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "STORE SUPPLY")) {
        printf(COLOR_BOLD_CYAN"\n>> STORE SUPPLY\033[0m\n\n");
        storesupply(&barang, &antrian, &nBarang);
        change = true;

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
}

```

```

else if (isKataSama(choice, "STORE REMOVE")) {
    printf(COLOR_BOLD_CYAN"\n>> STORE REMOVE\033[0m\n\n");
    storeremove(&barang, &nBarang);
    change = true;

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (startsWith(choice, "CART ADD")) {
    int i = 9;
    char namaBarang[MAX_LEN];
    int namaLength = 0;
    int jumlah = 0;

    while (i < choice.Length && choice.TabWord[i] == ' ') {
        i++;
    }

    int lastSpacePos = -1;
    for (int j = choice.Length - 1; j >= i; j--) {
        if (choice.TabWord[j] == ' ') {
            if (j + 1 < choice.Length &&
                choice.TabWord[j + 1] >= '1' &&
                choice.TabWord[j + 1] <= '9') {
                lastSpacePos = j;
                break;
            }
        }
    }
}

```

```

        if (lastSpacePos != -1) {
            for (int j = i; j < lastSpacePos; j++) {
                namaBarang[namaLength] = choice.TabWord[j];
                namaLength++;
            }
            namaBarang[namaLength] = '\\0';

            for (int j = lastSpacePos + 1; j < choice.Length; j++)
            {
                if (choice.TabWord[j] >= '1' && choice.TabWord[j]
<= '9') {
                    jumlah = jumlah * 10 + (choice.TabWord[j] -
'0');
                }
            }
        }

        if (lastSpacePos == -1) {
            printf(COLOR_BOLD_CYAN"\\n>> CART ADD\\033[0m\\n\\n");
            printf(COLOR_BOLD_RED"Gunakan format CART ADD <nama>
<jumlah> \\n"COLOR_OFF);
        } else {
            printf(COLOR_BOLD_CYAN"\\n>> CART ADD %s %d\\033[0m\\n\\n",
namaBarang, jumlah);
            cartAdd(&user, userIdx, barang, namaBarang, jumlah);
        }

        struct timespec req = {2, 0};

```

```

        nanosleep(&req, NULL);
    }
    else if (startsWith(choice, "CART REMOVE")) {
        int i = 12;
        char namaBarang[MAX_LEN];
        int namaLength = 0;
        int jumlah = 0;

        while (i < choice.Length && choice.TabWord[i] == ' ') {
            i++;
        }

        int lastSpacePos = -1;
        for (int j = choice.Length - 1; j >= i; j--) {
            if (choice.TabWord[j] == ' ') {
                if (j + 1 < choice.Length &&
                    choice.TabWord[j + 1] >= '1' &&
                    choice.TabWord[j + 1] <= '9') {
                    lastSpacePos = j;
                    break;
                }
            }
        }

        if (lastSpacePos != -1) {
            for (int j = i; j < lastSpacePos; j++) {
                namaBarang[namaLength] = choice.TabWord[j];
                namaLength++;
            }
        }
    }
}

```

```

        namaBarang[namaLength] = '\0';

        for (int j = lastSpacePos + 1; j < choice.Length; j++)
        {
            if (choice.TabWord[j] >= '1' && choice.TabWord[j]
<= '9') {
                jumlah = jumlah * 10 + (choice.TabWord[j] -
'0');
            }
        }
        if (lastSpacePos == -1) {
            printf(COLOR_BOLD_RED"Gunakan format CART REMOVE <nama>
<jumlah> \n"COLOR_OFF);
        } else {
            printf(COLOR_BOLD_CYAN"\n>> CART REMOVE %s
%d\033[0m\n\n", namaBarang, jumlah);
            cartRemove(&user, userIdx, namaBarang, jumlah);
        }

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "CART SHOW")) {
        printf(COLOR_BOLD_CYAN"\n>> CART SHOW\033[0m\n\n");
        cartShow(user, userIdx);

        struct timespec req = {4, 0};
        nanosleep(&req, NULL);
    }

```

```

    }
    else if(isKataSama(choice, "CART PAY")){
        printf(COLOR_BOLD_CYAN"\n>> CART PAY\033[0m\n\n");
        cartPay(&user, userIdx, &change);

        struct timespec req = {3, 0};
        nanosleep(&req, NULL);
    }
    else if (startsWith(choice, "HISTORY")) {
        int N = 0;
        int i = 8;
        int validInput = 1;

        while (i < choice.Length && choice.TabWord[i] == ' ') {
            i++;
        }

        if (i >= choice.Length) {
            printf(COLOR_BOLD_RED"\nMasukkan jumlah history yang
valid (bilangan bulat positif)!\n"COLOR_OFF);
            validInput = 0;
        } else {
            while (i < choice.Length && choice.TabWord[i] >= '0' &&
choice.TabWord[i] <= '9') {
                N = N * 10 + (choice.TabWord[i] - '0');
                i++;
            }

            while (i < choice.Length && choice.TabWord[i] == ' ') {

```

```

        i++;
    }

    if (i < choice.Length) {
        printf(COLOR_BOLD_RED"\nInput HISTORY harus berupa
bilangan bulat positif!\n"COLOR_OFF);
        validInput = 0;
    } else if (N <= 0) {
        printf(COLOR_BOLD_RED"\nJumlah history harus lebih
dari 0!\n"COLOR_OFF);
        validInput = 0;
    }
}

if (validInput) {
    printf(COLOR_BOLD_CYAN"\n>> HISTORY %d\033[0m\n\n", N);
    displayHistory(user, userIdx, N);
}
struct timespec req = {4, 0};
nanosleep(&req, NULL);
}
else if (isKataSama(choice, "WISHLIST ADD")) {
    printf("\n\033[1;34m>> WISHLIST ADD\033[0m\n");
    wishlistAdd(&user, &barang, userIdx);

    struct timespec req = {3, 0};
    nanosleep(&req, NULL);
}

```

```

else if (isKataSama(choice, "WISHLIST REMOVE")) {
    printf("\n\033[1;34m>> WISHLIST REMOVE\033[0m\n");
    wishlistRemove(&user, userIdx);

    struct timespec req = {3, 0};
    nanosleep(&req, NULL);
}
else if (startsWith(choice, "WISHLIST SWAP")) {
    int N = 0;
    int i = 14;
    int M = 0;

    while (i < choice.Length && choice.TabWord[i] == ' ') {
        i++;
    }

    while (i < choice.Length && choice.TabWord[i] >= '0' &&
choice.TabWord[i] <= '9') {
        N = N * 10 + (choice.TabWord[i] - '0');
        i++;
    }

    while (i < choice.Length && choice.TabWord[i] == ' ') {
        i++;
    }

    while (i < choice.Length && choice.TabWord[i] >= '0' &&
choice.TabWord[i] <= '9') {
        M = M * 10 + (choice.TabWord[i] - '0');

```



```

        i++;
    }

    if (N > 0 && M > 0) {
        printf(COLOR_BOLD_CYAN"\n>> WISHLIST SWAP %d
%d\n\n"COLOR_OFF, N, M);
        wishlistSwap(&user, userIdx, N, M);
    } else {
        printf(COLOR_BOLD_CYAN"\n>> WISHLIST
SWAP\n\n"COLOR_OFF);
        printf(COLOR_BOLD_RED"Gunakan format WISHLIST SWAP
<nomor1> <nomor2> \n"COLOR_OFF);
    }

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (startsWith(choice, "WISHLIST REMOVE")) { //ini remove
<i>

    int N = 0;
    int i = 16;

    while (i < choice.Length && choice.TabWord[i] == ' ') {
        i++;
    }

    while (i < choice.Length && choice.TabWord[i] >= '0' &&
choice.TabWord[i] <= '9') {
        N = N * 10 + (choice.TabWord[i] - '0');

```

```

        i++;
    }

    if (N > 0) {
        printf(COLOR_BOLD_CYAN"\n>> WISHLIST REMOVE
%d\n\n"COLOR_OFF, N);
        wishlistRemovei(&user, userIdx, N);
    } else {
        printf(COLOR_BOLD_CYAN"\n>> WISHLIST
REMOVE\033[0m\n\n");
        printf(COLOR_BOLD_RED"Gunakan format WISHLIST REMOVE
<nomor> \n"COLOR_OFF);
    }

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (isKataSama(choice, "WISHLIST CLEAR")) {
    printf(COLOR_BOLD_CYAN"\n>> WISHLIST CLEAR \n\n"COLOR_OFF);
    wishlistClear(&user, userIdx);

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (isKataSama(choice, "WISHLIST SHOW")) {
    printf(COLOR_BOLD_CYAN"\n>> WISHLIST SHOW \n\n"COLOR_OFF);
    wishlistShow(user, userIdx);

    struct timespec req = {3, 0};

```

```

        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "OPTIMASI RUTE")) {
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\n>> OPTIMASI RUTE \n\n"COLOR_OFF);
        optimasiRute();

        struct timespec req = {3, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "LOGOUT")) {
        printf(COLOR_BOLD_CYAN"\n>> LOGOUT\n\n"COLOR_OFF);
        logoutUser(&userIdx, user);

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
        loggedIn = false;
    }
    else if (startsWith(choice, "SAVE")) {
        char savefile[100] = "save/";
        int savefileLen = 5;
        int i = 5;

        while (i < choice.Length && choice.TabWord[i] == ' ') {
            i++;
        }

        while (i < choice.Length) {

```

```

        savefile[savefileLen++] = choice.TabWord[i];
        i++;
    }
    savefile[savefileLen] = '\\0';

    printf(COLOR_BOLD_CYAN"\\n>> SAVE %s\\033[0m\\n", savefile +
5);

    save(savefile, savefileLen, barang, user, nBarang, nUser);

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (isKataSama(choice, "QUIT")) {
    if (change){
        system("cls || clear");
        displayPurrMart();
        printf(COLOR_BOLD_CYAN"\\n>> QUIT\\n\\n"COLOR_OFF);
        quit(barang, user, nBarang, nUser, change);
    }
    else {
        printf(COLOR_BOLD_CYAN"\\n>> QUIT\\n\\n"COLOR_OFF);
        quit(barang, user, nBarang, nUser, change);
    }
}
else if (isKataSama(choice, "HELP")) {
    printf(COLOR_BOLD_CYAN"\\n>> HELP\\n\\n"COLOR_OFF);
    help3();

    struct timespec req = {4, 0};

```

```

        nanosleep(&req, NULL);
    }
    else {
        printf("Menu tidak valid, silahkan coba lagi\n");

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
}
return 0;
}

```

5 Algoritma-Algoritma Menarik

5.1 Algoritma Random Number Generator

Algoritma ini digunakan untuk memberikan rekomendasi random pada fitur ENHANCE. Algoritma ini memanfaatkan fungsi rand() yang ada pada stdlib.h, dimana rand() mengembalikan sebuah seed angka random. Hasil rand() kemudian dimodulasi dengan (batas bawah- batas atas + 1) kemudian dijumlah dengan batas bawah. rng() diimplementasikan dua kali dalam ENHANCE, yaitu untuk memilih album dan memilih lagu dalam album. Batas bawah dan batas atas yang dipilih adalah range banyak album an banyak lagu dalam album. Algoritma ini menarik karena dapat menghasilkan angka random yang selalu dapat menghasilkan sebuah lagu valid.

Algoritma ini digunakan untuk memberikan angka dan kata random yang digunakan pada fitur WORK CHALLENGE. Algoritma ini memanfaatkan fungsi rand() yang ada pada stdlib.h, dimana rand() mengembalikan sebuah seed angka random. Hasil rand() kemudian dimodulasi dengan 100 + 1. rng() diimplementasikan dua kali dalam WORK CHALLENGE, yaitu untuk mengambil angka random dan kata random. Algoritma ini menarik karena dapat menghasilkan angka random yang akan membuat challenge dari work menjadi lebih menarik.

5.2 Algoritma Delay Waktu

Algoritma delay waktu dalam main.c menggunakan fungsi `nanosleep()` dari pustaka `time.h` untuk memberikan jeda eksekusi program, yang memungkinkan pengguna untuk membaca pesan yang ditampilkan sebelum melanjutkan ke langkah berikutnya. Dengan mengatur waktu delay, seperti 2 atau 4 detik, program meningkatkan interaksi pengguna dan menghindari eksekusi yang terlalu cepat, sehingga pengguna dapat lebih mudah mengikuti alur program dan membuat keputusan yang tepat, seperti memilih menu atau memasukkan perintah. Hal ini menjadikan algoritma ini menarik karena tidak hanya berfungsi sebagai jeda, tetapi juga meningkatkan pengalaman pengguna secara keseluruhan.

5.3 Algoritma Pewarnaan Teks Terminal

Algoritma pewarnaan teks terminal menggunakan kode escape ANSI untuk mengubah tampilan teks dalam lingkungan terminal. Setiap definisi warna dan gaya merupakan urutan karakter khusus yang diinterpretasikan oleh terminal untuk mengubah properti tampilan teks. Kode-kode ini memungkinkan pengembang untuk memberikan visual yang lebih dinamis dan informatif dalam antarmuka berbasis teks.

Algoritma ini memanfaatkan urutan escape `\e[` diikuti dengan parameter numerik yang menentukan warna, ketebalan, dan efek tambahan seperti berkedip. Setiap kode memiliki fungsi spesifik:

- Warna dasar (`COLOR_CYAN`, `COLOR_OFF`): Mengubah warna teks atau mengembalikan ke pengaturan default
- Warna tebal (`COLOR_BOLD_RED`, `COLOR_BOLD_GREEN`): Membuat teks menjadi lebih tebal dan menonjol
- Efek berkedip (`COLOR_BLINK`, `COLOR_BOLD_CYAN_BLINK`): Menambahkan animasi berkedip pada teks

Algoritma ini menarik karena mampu meningkatkan keterbacaan dan estetika output terminal tanpa memerlukan pustaka eksternal, murni menggunakan dukungan bawaan sistem terminal. Penggunaan kode warna membantu pengguna dengan cepat mengidentifikasi jenis informasi, seperti pesan kesalahan (merah), keberhasilan (hijau), atau informasi penting (biru),

sehingga meningkatkan pengalaman pengguna dalam berinteraksi dengan antarmuka berbasis teks.

6 Data Test

6.1 PROFILE

Fitur : PROFILE

Hasil yang diharapkan :

```
>> PROFILE
Username : user1
Money : 100

Total Pengeluaran : 720
Total Pendapatan : 820

Daftar Barang yang Dimiliki:
- AK47 : 7
- Meong : 1
- Ayam Goreng Crisbar : 1
```

Hasil yang didapat :

```
>> PROFILE
Username : user1
Money : 100

Total Pengeluaran : 720
Total Pendapatan : 820

Daftar Barang yang Dimiliki:
- AK47 : 7
- Meong : 1
- Ayam Goreng Crisbar : 1
```

6.2 **CART ADD** <nama> <n>

Fitur : CART ADD <nama> <n>

Hasil yang diharapkan :

```
>> CART ADD AK47 20
```

Berhasil menambahkan 20 AK47 ke keranjang belanja!

```
>> CART ADD BebekKaliya 240
```

Barang tidak ada di toko!

Hasil yang didapat :

```
>> CART ADD AK47 20
```

Berhasil menambahkan 20 AK47 ke keranjang belanja!

```
>> CART ADD BebekKaliya 240
```

Barang tidak ada di toko!

6.3 **CART REMOVE** <nama> <n>

Fitur : CART REMOVE <nama> <n>

Hasil yang diharapkan :

```
>> CART REMOVE AK47 10
```

Berhasil mengurangi 10 AK47 dari keranjang belanja!

```
>> CART REMOVE AK47 70
```

Tidak berhasil mengurangi, hanya terdapat 10 AK47 pada keranjang!

```
>> CART REMOVE BintangSkibidi 70
```

Barang tidak ada di keranjang belanja!

Hasil yang didapat :

```
>> CART REMOVE AK47 10
```

Berhasil menghapus 10 AK47 dari keranjang belanja!

```
>> CART REMOVE AK47 70
```

Kuantitas barang yang ingin dihapus melebihi kuantitas barang di keranjang!

```
>> CART REMOVE BintangSkibidi 70
```

Barang tidak ditemukan di keranjang!

6.4 CART SHOW

Fitur : CART SHOW

Hasil yang diharapkan :

```
>> CART SHOW
```

Berikut adalah isi keranjangmu.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	20
---	--------	----

Total biaya yang harus dikeluarkan adalah 40.

```
>> CART SHOW
```

Keranjang kamu kosong!

Hasil yang didapat :

```
>> CART SHOW
```

Berikut adalah isi keranjangmu.

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Total biaya yang harus dikeluarkan adalah 40

```
>> CART SHOW
```

Keranjang kamu kosong!

6.5 CART PAY

Fitur : CART PAY

Hasil yang diharapkan :

```
>> CART PAY
```

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?
(Ya/Tidak): Ya

Selamat kamu telah membeli barang-barang tersebut!

```
>> CART PAY
```

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?
(Ya/Tidak): Ya

Uang kamu hanya 15, tidak cukup untuk membeli keranjang!

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	10
---	--------	----

Total biaya yang harus dikeluarkan adalah 30, apakah jadi dibeli?
(Ya/Tidak): **Tidak**

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	10
---	--------	----

Total biaya yang harus dikeluarkan adalah 30, apakah jadi dibeli?
(Ya/Tidak): **Purry**

>> **CART PAY**

Keranjang kamu kosong!

Hasil yang didapat :

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	20
---	--------	----

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?
(Ya/Tidak): Ya
Selamat kamu telah membeli barang-barang tersebut!

```
>> CART PAY
```

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?

(Ya/Tidak): Ya

Uang kamu hanya 15, tidak cukup untuk membeli keranjang!

```
>> CART PAY
```

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?

(Ya/Tidak): Tidak

```
>> CART PAY
```

Keranjang kamu kosong!

6.6 HISTORY <n> (Bonus)

Fitur : History <n> (Riwayat Maksimal)

Hasil yang diharapkan :

```
>> HISTORY 3
```

Riwayat pembelian barang:

Pembelian 1 - Total 40

Kuantitas	Nama	Total
2	AK47	20
1	Lalabu	20

Pembelian 2 - Total 100

Kuantitas	Nama	Total
-----------	------	-------

8	AK47	80
1	Lalabu	20
Pembelian 3 - Total 35		
Kuantitas	Nama	Total
3	AK47	30
1	Lalabu	20

Hasil yang didapat :

```
>> HISTORY 3
Riwayat pembelian barang:
Pembelian 1 - Total 40
Kuantitas Nama Total
2 AK47 20
1 Lalabu 20

Pembelian 2 - Total 100
Kuantitas Nama Total
8 AK47 80
1 Lalabu 20

Pembelian 3 - Total 50
Kuantitas Nama Total
3 AK47 30
1 Lalabu 20
```

6.7 WISHLIST ADD

Fitur : WISHLIST ADD

Hasil yang diharapkan :

```
>> WISHLIST ADD
Masukkan nama barang: Ayam Goreng Crisbar

Berhasil menambahkan Ayam Goreng Crisbar ke wishlist
```

```
>> WISHLIST ADD
```

Masukkan nama barang: **Ayam Goreng Crisbar**

Ayam Geprek Bakar Crispy sudah ada di wishlist!

```
>> WISHLIST ADD
```

Masukkan nama barang: **Ayam Bakar Crisbar**

Tidak ada barang dengan nama Ayam Bakar Crisbar!

Hasil yang didapat:

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Goreng Crisbar

Berhasil menambahkan **Ayam Goreng Crisbar** ke wishlist!

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Goreng Crisbar

Ayam Goreng Crisbar sudah ada di wishlist!

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Bakar Crisbar

Tidak ada barang dengan nama Ayam Bakar Crisbar!

6.8 WISHLIST SWAP <i> <j>

Fitur : WISHLIST SWAP <i> <j>

Hasil yang diharapkan :

```
>> WISHLIST SWAP 1 2
```

Berhasil menukar posisi Ayam Goreng Crisbar dengan AK47 pada wishlist!

// Wishlist terdefinisi pada kedua posisi

```
>> WISHLIST SWAP 1 2
```

Gagal menukar posisi Ayam Goreng Crisbar!

```
// Wishlist tidak terdefinisi pada salah satu posisi
```

```
>> WISHLIST SWAP 3 4
```

Posisi yang dimasukkan tidak valid!

```
// Wishlist tidak terdefinisi pada kedua posisi
```

```
>> WISHLIST SWAP X Y
```

Gunakan format WISHLIST SWAP <nomor1> <nomor2>

```
// Perintah tidak valid
```

Hasil yang didapat :

```
>> WISHLIST SWAP 1 2
```

Berhasil menukar posisi Ayam Goreng Crisbar dengan AK47 pada wishlist!

```
>> WISHLIST SWAP 1 2
```

Gagal menukar posisi Ayam Goreng Crisbar!

```
>> WISHLIST SWAP 3 4
```

Posisi yang dimasukkan tidak valid!

Masukkan perintah Anda: WISHLIST SWAP X Y

```
>> WISHLIST SWAP
```

Gunakan format WISHLIST SWAP <nomor1> <nomor2>

6.9 WISHLIST REMOVE <i>

Fitur : WISHLIST REMOVE <i>

Hasil yang diharapkan :


```
>> WISHLIST REMOVE 2
```

```
// Posisi wishlist terdefinisi
```

Berhasil menghapus barang posisi ke-2 dari wishlist!

```
>> WISHLIST REMOVE 10
```

```
// Posisi wishlist tidak terdefinisi
```

Penghapusan barang WISHLIST gagal dilakukan, Barang ke-10 tidak ada di WISHLIST!

```
>> WISHLIST REMOVE 1
```

```
// Wishlist kosong
```

Penghapusan barang WISHLIST gagal dilakukan, WISHLIST kosong!

```
>> WISHLIST REMOVE XY
```

```
// Perintah tidak valid
```

Gunakan format WISHLIST REMOVE <nomor>

Hasil yang didapat :

```
>> WISHLIST REMOVE 2
```

Berhasil menghapus barang posisi ke-2 di wishlist!

```
>> WISHLIST REMOVE 10
```

Penghapusan barang WISHLIST gagal dilakukan, Barang ke-10 tidak ada di WISHLIST!

```
>> WISHLIST REMOVE 1
```

Penghapusan barang WISHLIST gagal dilakukan, WISHLIST kosong!

Masukkan perintah Anda: WISHLIST REMOVE XY

```
>> WISHLIST REMOVE
```

Gunakan format WISHLIST REMOVE <nomor>

6.10 WISHLIST REMOVE

Fitur : WISHLIST REMOVE

Hasil yang diharapkan :

```
>> WISHLIST REMOVE
```

```
Masukkan nama barang yang akan dihapus : Lalabu
```

```
Lalabu berhasil dihapus dari WISHLIST!
```

```
>> WISHLIST REMOVE
```

```
Masukkan nama barang yang akan dihapus : LoremIpsum
```

```
Penghapusan barang WISHLIST gagal dilakukan, LoremIpsum tidak ada di  
WISHLIST!
```

Hasil yang didapat :

```
>> WISHLIST REMOVE
```

```
Masukkan nama barang yang akan dihapus : Lalabu
```

```
Lalabu berhasil dihapus dari WISHLIST!
```

```
>> WISHLIST REMOVE
```

```
Masukkan nama barang yang akan dihapus : LoremIpsum
```

```
Penghapusan barang WISHLIST gagal dilakukan, LoremIpsum tidak ada di WISHLIST!
```

6.11 WISHLIST CLEAR

Fitur : WISHLIST CLEAR

Hasil yang diharapkan :

```
>> WISHLIST CLEAR
```

```
Wishlist telah dikosongkan.
```

Hasil yang didapat :

```
>> WISHLIST CLEAR  
Wishlist telah dikosongkan.
```

6.12 WISHLIST SHOW

Fitur : WISHLIST SHOW

Hasil yang diharapkan :

```
>> WISHLIST SHOW  
// Wishlist terdefinisi  
Berikut adalah isi wishlist-mu:  
1 AK47  
2 Ayam Goreng Crisbar  
3 Lalabu
```

```
>> WISHLIST SHOW  
// Wishlist kosong  
Wishlist kamu kosong!
```

Hasil yang didapat :

```
>> WISHLIST SHOW  
Berikut adalah isi wishlist-mu:  
1 AK47  
2 Ayam Goreng Crisbar  
3 Lalabu
```

```
>> WISHLIST SHOW  
Wishlist kamu kosong!
```

7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1.	PROFILE	memeriksa apakah nama dan saldo dapat ter- <i>display</i> dengan benar	input command “PROFILE”	-	username, uang, total pendapatan, total pengeluaran serta daftar barang dapat ter- <i>display</i>	username, uang, total pendapatan, total pengeluaran serta daftar barang dapat ter- <i>display</i>
2.	CART ADD <nama> <n>	memeriksa apakah dapat menambahkan barang ke keranjang	input command “CART ADD <nama barang> <jumlah barang>”	Kondisi 1 : AK47 20 Kondisi 2 : BebekKaliya 240	Kondisi 1 : Berhasil menambahkan 20 AK47 ke keranjang Kondisi 2 : Gagal karena tidak ada barang di toko	Kondisi 1 : Berhasil menambahkan 20 AK47 ke keranjang Kondisi 2 : Gagal karena tidak ada barang di toko
3.	CART REMOVE <nama> <n>	memeriksa apakah dapat mengurangi kuantitas barang atau menghapus barang dari keranjang	input command “CART REMOVE <nama barang> <jumlah barang>”	Kondisi 1 : AK47 10 Kondisi 2 : AK47 70 (lebih) Kondisi 3 : BintangSkibidi 70	Kondisi 1 : Berhasil mengurangi jumlah barang Kondisi 2 : Pesan “Kuantitas melebihi barang yang di keranjang” Kondisi 3 : Gagal karena tidak ada barang di keranjang	Kondisi 1 : Berhasil mengurangi jumlah barang Kondisi 2 : Pesan “Kuantitas melebihi barang yang di keranjang” Kondisi 3 : Gagal karena tidak ada barang di keranjang

4.	CART SHOW	memeriksa apakah isi keranjang bisa ter- <i>display</i> dengan benar	input command “CART SHOW”	Kondisi 1 : Keranjang ada barang Kondisi 2 : Keranjang kosong	Kondisi 1 : Menampilkan detail isi keranjang serta total biaya Kondisi 2 : Pesan “Tidak ada barang di keranjang”	Kondisi 1 : Menampilkan detail isi keranjang serta total biaya Kondisi 2 : Pesan “Tidak ada barang di keranjang”
5.	CART PAY	memeriksa proses membeli sesuai isi keranjang dan sesuai dengan uang user.	input command “CART PAY” setelah itu input “Ya” atau “Tidak” untuk mengkonfirmasi pembelian	Kondisi 1 : “Ya” dengan uang cukup Kondisi 2 : “Ya” dengan uang kurang Kondisi 3 : “Tidak” atau input kata lain Kondisi 4 : Keranjang kosong	Kondisi 1 : uang berkurang dan barang di keranjang kembali kosong Kondisi 2 : gagal membayar, kondisi uang dan keranjang tetap Kondisi 3 : kembali ke menu Kondisi 4 : pesan “Keranjang kamu kosong!” dan kembali ke menu	Kondisi 1 : uang berkurang dan barang di keranjang kembali kosong Kondisi 2 : gagal membayar, kondisi uang dan keranjang tetap Kondisi 3 : kembali ke menu Kondisi 4 : pesan “Keranjang kamu kosong!” dan kembali ke menu

6.	HISTORY <n>	memeriksa apakah bisa menampilkan riwayat pembelian	input command “HISTORY <n>”	input “HISTORY 3”	menampilkan riwayat pembelian	menampilkan riwayat pembelian
7.	WISHLIST ADD	memeriksa apakah bisa menambahkan barang ke wishlist user	input command “WISHLIST ADD” setelah itu input nama barang	Kondisi 1 : Lalabu Kondisi 2 : Lalabu (lagi) Kondisi 3 : ALSTRUKD AT	Kondisi 1 : Lalabu berhasil ditambahkan ke wishlist Kondisi 2 : Lalabu sudah ada di wishlist Kondisi 3 : Tidak ada barang dengan nama ALSTRUKDA T	Kondisi 1 : Lalabu berhasil ditambahkan ke wishlist Kondisi 2 : Lalabu sudah ada di wishlist Kondisi 3 : Tidak ada barang dengan nama ALSTRUKDA T
8.	WISHLIST SWAP <i> <j>	memeriksa apakah bisa menukar posisi urutan barang di wishlist	input command “WISHLIST SWAP <posisi barang 1> <posisi barang 2>”	Kondisi 1 : barang relevan sesuai posisi Kondisi 2 : ada posisi yang tidak ada barangnya Kondisi 3: kedua posisi yang	Kondisi 1 : Berhasil menukar posisi barang Kondisi 2 : Menampilkan pesan salah satu barang gagal ditukar, command tidak berjalan	Kondisi 1 : Berhasil menukar posisi barang Kondisi 2 : Menampilkan pesan salah satu barang gagal ditukar, command tidak berjalan

				dimasukkan tidak terdefinisi Kondisi 4: masukkan tidak valid	Kondisi 3: Menampilkan pesan posisi tidak valid, command tidak berjalan Kondisi 4: Menampilkan pesan masukkan tidak valid, command tidak berjalan	Kondisi 3: Menampilkan pesan posisi tidak valid, command tidak berjalan Kondisi 4: Menampilkan pesan masukkan tidak valid, command tidak berjalan
9.	WISHLIST REMOVE <i>	memeriksa apakah bisa menghapus barang berdasarkan posisi dari wishlist	input command “WISHLIST REMOVE <posisi barang>”	Kondisi 1 : “2” Kondisi 2 : “10”, barang ke-10 tidak ada Kondisi 3 : wishlist kosong Kondisi 4: masukkan tidak valid	Kondisi 1 : berhasil menghapus barang di posisi 2 dari wishlist Kondisi 2 : Gagal dan menampilkan pesan error bahwa barang ke-10 tidak ada Kondisi 3 : Gagal dan menampilkan pesan error bahwa wishlist kosong	Kondisi 1 : berhasil menghapus barang di posisi 2 dari wishlist Kondisi 2 : Gagal dan menampilkan pesan error bahwa barang ke-10 tidak ada Kondisi 3 : Gagal dan menampilkan pesan error bahwa wishlist kosong

					Kondisi 4: Gagal dan menampilkan pesan error bahwa masukkan tidak valid	Kondisi 4: Gagal dan menampilkan pesan error bahwa masukkan tidak valid
10.	WISHLIST REMOVE	memeriksa apakah bisa menghapus barang berdasarkan nama barang dari wishlist	input command “WISHLIST REMOVE” setelah itu input nama barang	Kondisi 1 : Lalabu Kondisi 2 : LoremIpsum	Kondisi 1 : Lalabu berhasil terhapus dari wishlist Kondisi 2 : Gagal karena barang tidak ada di wishlist	Kondisi 1 : Lalabu berhasil terhapus dari wishlist Kondisi 2 : gagal karena barang tidak ada di wishlist
11.	WISHLIST CLEAR	memeriksa apakah dapat menghapus seluruh barang di wishlist	input command “WISHLIST CLEAR”	-	Wishlist berhasil dikosongkan	Wishlist berhasil dikosongkan
12.	WISHLIST SHOW	memeriksa untuk menampilkan isi wishlist sesuai urutan	input command “WISHLIST SHOW”	Kondisi 1 : Wishlist terisi Kondisi 2 : Wishlist kosong	Kondisi 1 : menampilkan isi wishlist sesuai urutan Kondisi 2 : menampilkan pesan wishlist kosong	Kondisi 1 : menampilkan isi wishlist sesuai urutan Kondisi 2 : menampilkan pesan wishlist kosong

8 Pembagian Kerja dalam Kelompok

Ahmad Evander Ruizhi Xavier / 18223064	<ul style="list-style-type: none">- Wishlist Swap <nomor1> <nomor2>- Wishlist Remove <nomor>- Wishlist Clear- Wishlist Show
Nazwan Siddqi Muttaqin / 18223066	<ul style="list-style-type: none">- profile- cart add <nama> <n>- cart show- history <n>- bonus - Riwayat Maksimal- bonus - Optimasi Rute Ekspedisi- main
Keane Putra Lim / 18223056	<ul style="list-style-type: none">- wishlist add- wishlist remove
Sebastian Albern Nugroho / 18223074	<ul style="list-style-type: none">- Store List- ADT
Joan Melkior Silaen / 18223102	<ul style="list-style-type: none">- Cart remove- Cart pay

9 Lampiran

9.1 Deskripsi Tugas Besar

Buatlah sebuah aplikasi simulasi berbasis CLI (*command-line interface*). Sistem ini dibuat dalam **bahasa C** dengan menggunakan **struktur data yang sudah kalian pelajari** di

mata kuliah ini. Kalian boleh menggunakan (atau memodifikasi) struktur data yang sudah kalian buat untuk praktikum pada tugas besar ini. Daftar ADT yang wajib digunakan dapat dilihat pada bagian Daftar ADT. *Library* yang boleh digunakan hanya **stdio.h**, **stdlib.h**, **time.h**, dan **math.h**.

System Mechanic

1. About the System

PURRMART adalah sebuah aplikasi yang dapat mensimulasikan aktivitas beli barang pada *e-commerce*. PURRMART memiliki beberapa fitur utama, yaitu:

- Menampilkan barang toko
- Meminta dan menyuplai barang baru ke toko
- Menyimpan dan membeli barang dalam keranjang
- Menampilkan barang yang sudah dibeli
- Membuat dan menghapus *wishlist*
- Bekerja untuk menghasilkan uang

2. Menu Program

Ketika program pertama kali dijalankan, PURRMART akan memperlihatkan *main menu* yang berisi *welcome menu* dan beberapa *command* yaitu **START**, **LOAD**, dan juga **HELP**.

Setelah itu, program akan memasuki *login menu* yang memiliki command **LOGIN**, **REGISTER**, dan juga **HELP**. Jika pengguna berhasil memasuki kredensial suatu akun, maka mereka akan masuk ke menu selanjutnya.

Main menu menerima masukan berupa *command* yang akan dijelaskan pada bagian berikutnya. Program akan terus menerima *command* sampai diberikan *command* **QUIT** yang berlaku pada seluruh menu.

3. Command

Pengguna dapat memasukkan *command-command* berikut. Seluruh *command* hanya berlaku pada menu utama.

a. PROFILE

STEI- ITB	<nomor dokumen>	Halaman 49 dari 58 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

PROFILE adalah *command* yang digunakan untuk melihat data diri pengguna. PROFILE hanya dapat dipanggil saat status pengguna telah login

b. CART ADD <nama> <n>

CART ADD adalah *command* yang digunakan untuk menambahkan barang dengan kuantitas tertentu ke dalam keranjang belanja.

c. CART REMOVE <nama> <n>

CART REMOVE adalah *command* yang digunakan untuk mengurangi barang sejumlah kuantitas tertentu dari keranjang belanja. Perlu dilakukan validasi terhadap kuantitas yang diberikan, bila kuantitas pada keranjang belanja lebih sedikit dari N maka perintah akan gagal.

d. CART SHOW

CART SHOW adalah *command* yang digunakan untuk menunjukkan barang-barang yang sudah dimasukkan ke dalam keranjang.

e. CART PAY

CART PAY adalah *command* yang digunakan untuk membeli barang-barang yang sudah dimasukan ke dalam keranjang. Perlu dipastikan bahwa **pengguna memiliki uang yang cukup** untuk membeli seluruh barang keranjang. Pembelian akan mengurangi uang yang dimiliki pengguna dan menambahkan riwayat pembelian.

Nama barang yang dimasukan ke riwayat pembelian adalah barang dengan total harga (harga barang * kuantitas) terbesar. Jika terdapat lebih dari 1 barang dengan total yang sama, maka yang disimpan adalah barang dengan urutan lexical yang lebih besar. Dimasukan juga total harga pada pembelian tersebut. Jika terdapat barang dengan nama

Zebra dan AK47 yang memiliki total sama, maka Zebra akan dimasukan ke *history* karena secara lexical $z > a$.

Jika tidak mengerjakan bonus, maka barang yang dimasukan ke history **hanya 1** dengan ketentuan di atas.

f. HISTORY <n>

HISTORY adalah *command* yang digunakan untuk menunjukan riwayat pembelian seorang pengguna. N merupakan jumlah riwayat yang ditampilkan, contoh N=3 maka akan menampilkan 3 riwayat pembelian terbaru. Jika N melebihi jumlah riwayat pembelian yang ada, maka seluruh riwayat pembelian akan ditampilkan. Urutan penunjukan adalah dari yang paling baru ke paling tua.

g. WISHLIST ADD

WISHLIST ADD merupakan *command* yang digunakan untuk menambahkan suatu barang ke *wishlist*.

h. WISHLIST SWAP <i> <j>

WISHLIST SWAP merupakan *command* yang digunakan untuk menukar barang posisi ke-i dengan barang posisi ke-j pada *wishlist*. Posisi i dan j merupakan urutan barang pada *wishlist*, urutan dimulai dari 1.

i. WISHLIST REMOVE <i>

WISHLIST REMOVE adalah *command* yang digunakan untuk menghapus barang dengan posisi ke-i dari *wishlist*.

j. WISHLIST REMOVE

WISHLIST REMOVE adalah *command* yang digunakan untuk menghapus barang dari *wishlist* berdasarkan nama barang yang dimasukkan pengguna.

k. WISHLIST CLEAR

WISHLIST CLEAR adalah *command* yang digunakan untuk menghapus semua barang yang terdapat di dalam WISHLIST.

l. WISHLIST SHOW

WISHLIST SHOW adalah *command* yang digunakan untuk menunjukkan barang-barang yang sudah dimasukkan ke dalam wishlist.

4. Perubahan Command

Terdapat beberapa *command* iterasi sebelumnya yang berubah, yaitu sebagai berikut.

a. START, LOAD, dan SAVE

Terdapat perubahan konfigurasi yang harus ditambahkan dalam implementasi *command* START, LOAD, dan SAVE. **BACA KONFIGURASI SISTEM UNTUK PEMBAHARUAN SAVE FILE!**

b. STORE LIST

STORE LIST akan menampilkan nama barang yang dijual **beserta harganya**


9.2 Notulen Rapat

No. Kelompok/Kelas	: 07/02
Nama Kelompok	: 07yes
Anggota Kelompok (Nama/NIM)	: 1. Ahmad Evander Ruizhi X. (18223064) 2. Nazwan Siddqi Muttaqin (18223066) 3. Keane Putra Lim (18223056) 4. Sebastian Albern Nugroho (18223074) 5. Joan Melkior Silaen (18223102)

Asisten Pembimbing

: Aulia Nadhirah Yasmin B (18221066)

Asistensi I



Tanggal : 18 Desember 2024	Catatan Asistensi: Asistensi pertama untuk milestone 2 ini membahas mengenai <i>update progress</i> sejauh ini, penulisan laporan, teknis demo tubes, serta beberapa hal teknis lainnya. <ol style="list-style-type: none">1. Laporan milestone 2 dibedakan dengan laporan milestone 1. Isi dari laporan kali ini hanya membahas mengenai hal-hal yang dibuat di milestone 2.2. Demo alstrukdat akan dilakukan di kemudian hari, dan teknisnya kurang lebih akan sama dengan demo dari tubes daspro yang telah dilakukan di semester 2.3. Error handling harus ada namun outputnya dibebaskan.4. Fungsi load dan save masih error. Kemungkinan ada alokasi memori yang masih salah. <i>Review</i> ulang kodenya.
Tempat : Zoom Meeting	
Kehadiran Anggota Kelompok: 1. Ahmad Evander Ruizhi Xavier (18223064)  2. Nazwan Siddqi Muttaqin (18223066)	



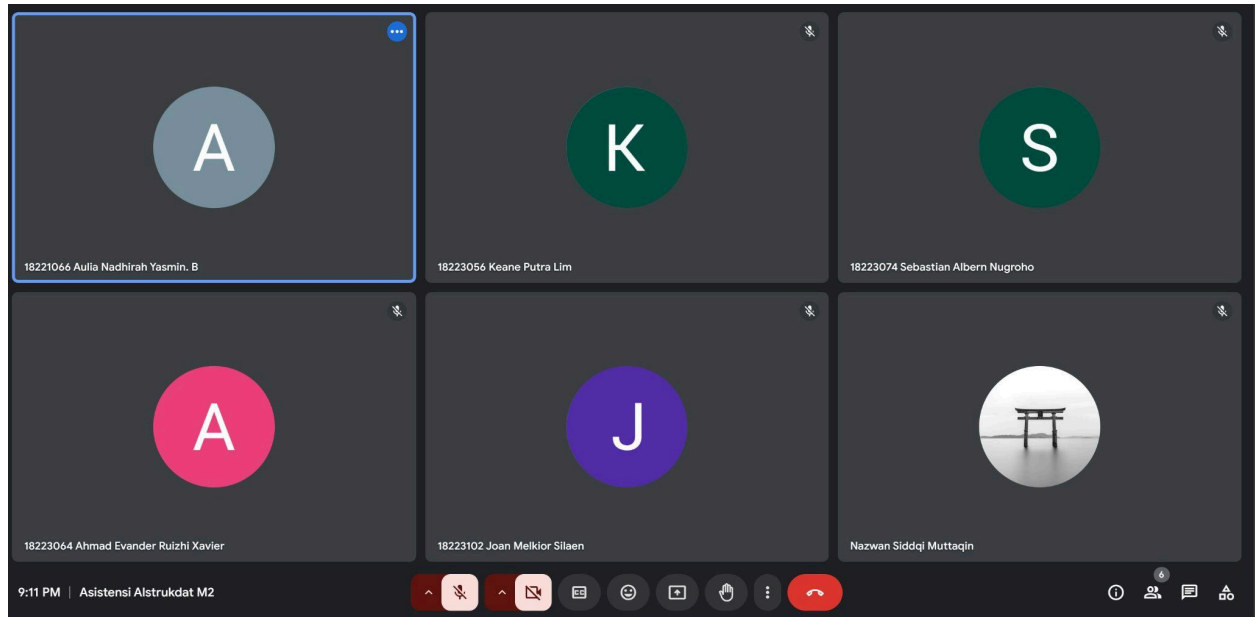
3. Sebastian Albern Nugroho (18223074)

4. Joan Melkior Silaen (18223102)

5. Keane Putra Lim (18223056)

	
	<p>Tanda Tangan Asisten:</p> 

Dokumentasi:



9.3 Log Activity Anggota Kelompok

Tanggal	NIM	Lokasi	Aktivitas
13/12/2024	18223064 18223066 18223056 18223074 18223102	Kost	Pembagian tugas awal.
15/12/2024	18223064	Kost	Penambahan ADT dan penyesuaian pada fungsi ML 1.
16/12/2024	18223066	Rumah	Penambahan profile dan optimalisasi readTxt.
17/12/2024	18223064	Kost	Penambahan fungsi wishlist swap, wishlist remove <i>,</i>

			wishlist clear, dan wishlist show
17/12/2024	18223066	Rumah	Penambahan fungsi history, cartAdd, cartShow, dan bonus riwayat maksimal.
18/12/2024	18223102	Kost	Penambahan fungsi cartRemove dan cartPay.
17/12/2024	18223074	Kost	Penambahan harga barang pada fungsi storeList.
18/12/2024	18223074	Kost	Merapikan folder ADT dan menyesuaikan directory “include”.
18/12/2024	18223066	Cafe	Merge semua fungsi yang telah dibuat oleh kelompok ke dalam main.
18/12/2024	18223064 18223066 18223056 18223074 18223102	Zoom Meeting	Asistensi 1
20/12/2024	18223064	Rumah	Memperbaiki pembacaan file pada readTxt untuk menyesuaikan konfigurasi save terbaru

20/12/2024	18223074	Kost	Merapikan directory pada fungsi wishlist dan cart, serta mulai mengerjakan driver untuk ADT yang baru ditambahkan.
21/12/2024	18223066	Rumah	Penambahan bonus optimasi rute dan finishing akhir.
21/12/2024	18223074	Rumah	Menyelesaikan driver untuk ADT yang baru ditambahkan.
22/12/2024	18223064	Rumah	Menambahkan beberapa error handling di bagian wishlist, menyesuaikan tampilan warna di beberapa pesan error
22/12/2024	18223066	Rumah	Finishing dengan memperbaiki error handling dan perubahan makefile.
17/12/2024	1822356	Kost	Mengerjakan wishlist add dan wishlist remove
21/12/2024	18223056	Kost	Laporan bagian ADT
22/12/2024	18223064 18223066 18223056 18223074 18223102	Kost dan Rumah Masing-Masing	Finish Milestone 2