

# 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
		IF2111-TB-02-07		51
		Revisi	<no revisi>	25-11-2024

# Daftar Isi

<b>1</b>	<b>Ringkasan.....</b>	<b>3</b>
<b>2</b>	<b>Penjelasan Tambahan Spesifikasi Tugas.....</b>	<b>5</b>
	2.1 Spesifikasi Fitur Save.....	5
<b>3</b>	<b>Struktur Data (ADT).....</b>	<b>5</b>
	3.1 Mesinkata.....	5
	3.2 Mesinkarakter.....	8
	3.3 Queue.....	9
	3.4 Boolean.....	11
	3.5 Arraydin.....	11
	3.7 ReadTxt.....	17
	3.8 Addon.....	18
	3.9 Custom.....	18
<b>4</b>	<b>Program Utama.....</b>	<b>18</b>
<b>5</b>	<b>Algoritma-Algoritma Menarik.....</b>	<b>24</b>
	5.1 Algoritma Random Number Generator.....	24
	5.2 Algoritma Delay Waktu.....	24
<b>6</b>	<b>Data Test.....</b>	<b>25</b>
	6.1 START.....	25
	6.2 LOAD.....	25
	6.3 REGISTER.....	26
	6.4 LOGIN.....	26
	6.5 LOGOUT.....	27
	6.6 WORK.....	27
	6.7 WORK CHALLENGE.....	28
	6.8 STORE LIST.....	31
	6.9 STORE REQUEST.....	32
	6.10 STORE SUPPLY.....	33
	6.11 STORE REMOVE.....	34
	6.12 HELP.....	34
	6.13 SAVE.....	36
	6.14 QUIT.....	37
<b>7</b>	<b>Test Script.....</b>	<b>38</b>

<b>8</b>	<b>Pembagian Kerja dalam Kelompok.....</b>	<b>42</b>
<b>9</b>	<b>Lampiran.....</b>	<b>43</b>
	9.1 Deskripsi Tugas Besar.....	43
	9.2 Notulen Rapat.....	47
	9.3 Log Activity Anggota Kelompok.....	49

# 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.

## **3 Struktur Data (ADT)**

### **3.1 Mesinkata**

ADT ini merupakan lanjutan dari ADT Mesinkarakter. ADT ini membaca input per kata dan menangani pemisahan kata. Program ini bisa menggantikan peran scanf. Program ini memiliki pembatas panjang kata berupa NMax untuk mencegah buffer overflow. Fungsi Mesinkata memiliki beberapa fungsi seperti:

#### **1. void IgnoreBlanks()**

Prosedur ini akan mengabaikan spasi kosong dalam aliran input. Prosedur ini terus memajukan pembacaan karakter sampai menemukan karakter selain spasi atau MARK. Prosedur ini membaca karakter satu per satu menggunakan ADV() hingga karakter yang sedang diakses alias currentChar.

#### **2. void IgnoreBlanksFile()**

Prosedur ini mengabaikan spasi dan karakter baris baru dalam file hingga menemukan karakter selain spasi atau MARK. Perbedaan antara fungsi ini dengan IgnoreBlanks adalah fungsi ini dirancang khusus untuk input dari file.

### 3. void STARTWORD()

Prosedur ini menginisialisasi mesin kata untuk membaca dari input standar. Program akan memanggil START() dan memanggil IgnoreBlanks() untuk mengabaikan spasi. Jika MARK ditemukan maka EndWord akan dinyatakan sebagai True. Jika tidak maka mesin akan memproses kata menggunakan ADVWORD()

### 4. void STARTWORDFILE(FILE \*inputFile)

Prosedur ini bertujuan untuk menginisialisasi mesin kata untuk membaca dari sebuah file tertentu. Prosedur ini menggunakan STARTFILE() untuk mempersiapkan mesin karakter dengan file yang diberikan. Selanjutnya program memanggil IgnoreBlanksFile() untuk melewati spasi dan baris baru. Jika MARK ditemukan maka EndWord menjadi True. Jika tidak maka prosedur ADVWORDFILE() akan memproses kata pertama

### 5. void ADVWORD()

Prosedur ini melanjutkan mesinkata ke kata berikutnya dalam input standar. Prosedur ini memanggil IgnoreBlanks() untuk melewati spasi sebelum kata berikutnya. Jika MARK ditemukan maka EndWord diatur menjadi True. Jika tidak maka kata berikutnya diambil menggunakan CopyWord()

### 6. void ADVWORDSpasi()

Prosedur bertujuan untuk memproses kata berikutnya dengan tetap menyimpan spasi dalam kata. Ini sangat berguna ketika memproses nama atau frasa di mana spasi internal bermakna. Berbeda dengan perpindahan kata biasa, prosedur ini tidak memperlakukan spasi sebagai pemisah kata kecuali jika itu adalah spasi di awal.

#### 7. void ADVWORDFILE()

Prosedur ini melanjutkan mesin kata ke kata berikutnya dalam sebuah file. Cara program ini bekerja seperti ADVWORD() tetapi menggunakan IgnoreBlanksFile() untuk melewati spasi dan baris baru dalam file. Kata berikutnya kemudian diambil menggunakan CopyWordFile()

#### 8. void CopyWord()

Prosedur ini bertujuan untuk menyalin kata ke dalam struktur CurrentWord dari input standar. Prosedur ini membaca karakter satu per satu dan menyimpannya di CurrentWord.TabWord hingga menemukan tanda akhir atau mencapai batas panjang kata maksimum NMax.

#### 9. void CopyWordFile()

Prosedur ini akan menyalin sebuah kata ke dalam struktur CurrentWord dari sebuah file. Berbeda seperti CopyWord(), program ini menangani karakter baris baru \n selain spasi dan tanda akhir. Prosedur memastikan bahwa akuisisi kata dapat bekerja dengan benar

#### 10. void CopyWordSpasi()

Prosedur ini bertujuan untuk menyalin kata ke dalam struktur CurrentWord. Ini mirip seperti CopyWord() tetapi tidak melewati spasi dalam kata. Hal ini memungkinkan nama atau frasa dengan spasi untuk disimpan dengan benar.

#### 11. boolean isKataSama(Word kata1, char\* kata2)

Fungsi ini memiliki tujuan untuk membandingkan kata untuk menentukan kesamaan. Fungsi ini menghitung panjang dari kata 1 dan kata 2. Jika panjangnya berbeda maka fungsi langsung mengembalikan false. Jika panjang dari kata1 dan kata 2 sama, maka hal selanjutnya yang akan dibandingkan adalah karakter. Jika semua karakter cocok maka fungsi akan mengembalikan true.

### **3.2 Mesinkarakter**

ADT ini digunakan untuk membaca input karakter per karakter dari sebuah kata / stdin (standard input). Fungsi ini akan lanjut membaca huruf per huruf pada sebuah kata sampai akhir

Terdapat beberapa fungsi pada file mesinkarakter seperti :

1. void START()

Fungsi ini bertujuan untuk mempersiapkan mesin untuk mulai membaca karakter dengan standard input stdin.

2. void STARTFILE(FILE \*inputFile)

Prosedur ini bertujuan untuk mempersiapkan mesin untuk membaca karakter pada suatu file tertentu. Prosedur ini mirip seperti START() tetapi memiliki perbedaan. Perbedaan berupa prosedur ini membaca file yang di diberikan pada input file daripada membaca dari standard input

3. void ADV()

Prosedur ini bertujuan untuk membaca karakter selanjutnya pada suatu kata. Jika akhiran menentukan bahwa EOP adalah true, maka pita akan ditutup yang bisa dilihat pada kode fclose(pita)

4. void READADV()

Prosedur ini mirip seperti ADV(), tetapi memiliki konsiderasi null '\0' sebagai akhiran ada suatu input. ADV() menggunakan MARK untuk mengidentifikasi sebagai akhiran dari sebuah input sedangkan READADV() menggunakan karakter null



#### 5. char GetCC()

Fungsi ini mengembalikan karakter yang sedang dibaca pada suatu kata.

#### 6. boolean IsEOP()

Fungsi ini mengecek jika akhiran dari suatu input sudah ditemui/tercapai.

### 3.3 Queue

ADT Queue merupakan ADT yang cocok digunakan untuk implementasi antrian, buffer data, manajemen proses dan perencanaan tugas. Karakteristik dari queue adalah FIFO alias First In First Out. Konsep ini sama seperti seseorang sedang mengantri untuk membeli barang. Bagi orang yang mengantri dahulu, orang tersebut akan dilayani dahulu. Dalam konteks programming, penambahan dilakukan pada tail dan pengambilan dilakukan di head. Fungsi fungsi di dalam Queue berupa:

#### 1. void CreateQueue(Queue \*q)

Prosedur ini akan menciptakan sebuah queue yang baru dan kosong. Pada saat ini, queue masih kosong dan semua elemen pada queue diatur menjadi nilai `IDX_UNDEF`

#### 2. boolean isEmptyQueue(Queue q)

Fungsi `isEmptyQueue` akan memeriksa jika queue kosong. Output yang dihasilkan untuk memberitahu status kapasitas queue berupa boolean

#### 3. boolean isFullQueue(Queue q)

Fungsi ini memeriksa jika queue sudah mencapai kapasitas maksimal. Boolean berupa `true` atau `false` akan dikeluarkan untuk menandakan status kapasitas queue

#### 4. int lengthQueue(Queue q)

Prosedur ini menghitung jumlah elemen dalam queue. Prosedur akan mengembalikan 0 jika queue kosong dan panjang queue akan dihitung berdasarkan posisi idxHead dan idxTail jika terdapat elemen

5. void enqueue(Queue \*q, ElTypeQ val)

Prosedur enqueue menambahkan elemen baru ke bagian belakang queue (TAIL). Jika queue kosong, idxHead dan idxTail diatur ke 0. Jika tidak, idxTail digeser melingkar menggunakan  $(\text{idxTail} + 1) \% \text{CAPACITY}$ . Lalu, elemen baru dimasukkan di posisi TAIL.

6. void dequeue(Queue \*q, ElTypeQ \*val)

Prosedur dequeue menghapus elemen pada bagian depan queue berupa HEAD. Karena ini berupa prosedur, maka tidak ada nilai yang akan dikembalikan oleh prosedur. Elemen Head dihapus dan disimpan ke parameter val. Jika elemen yang dihapus adalah satu-satunya elemen, idxHead dan idxTail dikembalikan ke IDX\_UNDEF. Jika tidak, idxHead digeser melingkar menggunakan  $(\text{idxHead} + 1) \% \text{CAPACITY}$

7. void displayQueue(Queue q)

Prosedur ini tidak mengembalikan nilai karena berupa void. Jika queue kosong maka prosedur akan menampilkan [] sedangkan jika queue memiliki isi maka program akan menampilkan elemen dari HEAD sampai TAIL

8. boolean searchQueue(Queue q, ElTypeQ val)

Fungsi ini mampu mencari elemen tertentu dalam queue berdasarkan nilai. Fungsi ini mengembalikan true jika elemen dengan nilai yang sesuai ditemukan dalam queue. Selain itu, fungsi juga akan memeriksa setiap elemen dalam buffer hingga idxTail

### 3.4 Boolean

ADT ini menyediakan tipe data boolean yang konsisten. ADT ini memudahkan dalam penggunaan nilai true / false. Keunggulan dari ADT ini berupa penggunaan memori yang minimal sehingga operasi boolean menjadi efisien dan cepat.

### 3.5 Arraydin

ADT Array Din merupakan ADT untuk mengelola memori dinamis. Program ini menggunakan malloc() untuk alokasi awal dan membebaskan memori dengan free() saat tidak digunakan. Ukuran dari memori dapat berubah sesuai kebutuhan. Keunggulan dari ADT ini adalah program ini mampu mengakses elemen dengan cepat mengingat time-complexity berupa  $O(1)$ . Selanjutnya, ukuran dari memori dinamis sehingga mampu beradaptasi dengan kebutuhan permasalahan.

#### 1. ArrayDin MakeArrayDin()

Fungsi MakeArrayDin bertipe ArrayDin digunakan untuk membuat array dinamis baru. Fungsi ini mengalokasikan memori sebesar InitialSize, menginisialisasi kapasitas array menjadi InitialSize, dan menetapkan jumlah elemen efektif (Neff) menjadi nol. Array dinamis ini kemudian dikembalikan sebagai hasil fungsi.

#### 2. void DeallocateArrayDin(ArrayDin \*array)

Fungsi DeallocateArrayDin bertipe void digunakan untuk membebaskan memori yang telah dialokasikan untuk array dinamis. Fungsi ini memastikan bahwa pointer A di dalam array terdelokasi, sehingga mencegah kebocoran memori.

#### 3. boolean isEmpty(ArrayDin array)

Fungsi isEmpty bertipe boolean digunakan untuk memeriksa apakah array dinamis kosong. Fungsi ini akan mengembalikan nilai true jika jumlah elemen efektif (Neff) bernilai nol, dan false jika sebaliknya.

#### 4. int length(ArrayDin array)

Fungsi length bertipe int digunakan untuk mendapatkan jumlah elemen efektif dalam array. Fungsi ini mengembalikan nilai Neff, yang merepresentasikan banyaknya elemen yang saat ini berada dalam array.

#### 5. elType get(ArrayDin array, idxType i)

Fungsi get bertipe elType digunakan untuk mendapatkan elemen tertentu di indeks logis ke-i dalam array. Fungsi ini mengembalikan nilai elemen pada indeks yang diminta, dengan asumsi indeks berada dalam rentang yang valid dan array tidak kosong.

#### 6. int getCapacity(ArrayDin array)

Fungsi getCapacity bertipe int digunakan untuk mengembalikan kapasitas total array. Kapasitas ini menunjukkan jumlah elemen maksimum yang dapat ditampung oleh array tanpa perlu dialokasikan ulang.

#### 7. void insertAt(ArrayDin \*array, elType el, idxType i)

Fungsi insertAt bertipe void digunakan untuk menambahkan elemen baru pada indeks tertentu dalam array. Jika kapasitas array penuh, fungsi ini akan menggandakan kapasitasnya dengan

mengalokasikan ulang memori. Setelah itu, elemen-elemen di belakang indeks  $i$  akan digeser untuk memberikan ruang bagi elemen baru, dan  $Neff$  akan diperbarui.

#### 8. `void deleteAt(ArrayDin *array, idxType i)`

Fungsi `deleteAt` bertipe `void` digunakan untuk menghapus elemen pada indeks tertentu dari array. Fungsi ini menggeser elemen-elemen setelah indeks  $i$  ke posisi sebelumnya untuk menutup celah, dan kemudian mengurangi nilai  $Neff$ .

#### 9. `idxType search(ArrayDin array, elType el)`

Fungsi `search` bertipe `idxType` digunakan untuk mencari elemen tertentu di dalam array berdasarkan atribut tertentu (misalnya, nama elemen). Jika elemen ditemukan, fungsi ini akan mengembalikan indeks elemen tersebut. Jika elemen tidak ditemukan, fungsi ini mengembalikan nilai  $-1$  sebagai indikasi kegagalan.

### 3.6 List

ADT List berguna untuk menyimpan data sekuensial dan juga implementasi list dengan ukuran maksimum yang diberikan. Kompleksitas yang dilakukan ADT ini untuk mengakses elemen berupa  $O(1)$  sedangkan kompleksitas pencarian, penyisipan/penghapusan dan juga penggabungan sebesar  $O(n)$ . ADT ini cocok digunakan ketika program membutuhkan akses dengan cepat dan sering melakukan insert/delete

#### 1. List `MakeList()`

Fungsi ini digunakan untuk menginisialisasi sebuah list kosong. Semua elemen array dalam list akan diatur ke nilai penanda (Mark) sebagai indikator bahwa elemen tersebut belum digunakan. Proses ini dilakukan dengan iterasi melalui seluruh indeks dari  $0$  hingga kapasitas maksimum ( $MaxEl$ ) list. Fungsi ini memastikan bahwa list dalam keadaan kosong sebelum mulai digunakan.

## 2. boolean IsEmpty(List L)

Fungsi ini memeriksa apakah list dalam keadaan kosong dengan melihat elemen pada indeks pertama. Jika elemen tersebut memiliki nilai Mark, maka fungsi akan mengembalikan nilai true, menandakan list kosong. Jika elemen pertama tidak bernilai Mark, maka list dianggap memiliki elemen efektif dan fungsi mengembalikan nilai false.

## 3. ElType Get(List L, IdxType i)

Fungsi ini digunakan untuk mengakses elemen pada indeks tertentu dalam list. Fungsi mengembalikan elemen yang tersimpan pada indeks ke-i tanpa melakukan perubahan pada list. Hal ini mempermudah pengambilan data secara langsung berdasarkan indeks tertentu.

## 4. void Set(List \*L, IdxType i, ElType v)

Fungsi ini berfungsi untuk memperbarui elemen pada indeks ke-i dengan elemen baru v. Fungsi mengubah elemen dalam list secara langsung, memungkinkan modifikasi data pada posisi tertentu tanpa mempengaruhi elemen lain.

## 5. int Length(List L)

Fungsi ini menghitung jumlah elemen efektif dalam list, yaitu elemen yang tidak bernilai Mark. Prosesnya dilakukan dengan iterasi melalui seluruh elemen list dan mengakumulasi jumlah elemen yang valid. Hasil akhirnya memberikan panjang aktual list.

## 6. IdxType FirstIdx(List L)

Fungsi ini mengembalikan indeks pertama dari list, yaitu index 0. Namun, jika list kosong (tidak ada elemen efektif), fungsi akan mengembalikan nilai InvalidIdx, menandakan bahwa tidak ada elemen dalam list.

#### 7. IdxType LastIdx(List L)

Fungsi ini mengembalikan indeks terakhir dari elemen efektif dalam list. Indeks terakhir dihitung dengan mengurangi 1 dari panjang list yang diperoleh dari fungsi Length. Jika list kosong, fungsi tidak akan memberikan nilai indeks yang valid.

#### 8. boolean IsIdxValid (List L, IdxType i)

Fungsi ini memeriksa apakah sebuah indeks berada dalam rentang valid array, yaitu antara 0 hingga MaxEl-1. Fungsi ini digunakan untuk memastikan bahwa operasi terhadap list tidak keluar dari batas array yang tersedia.

#### 9. boolean IsIdxEff (List L, IdxType i)

Fungsi ini memastikan bahwa indeks yang diberikan berada dalam rentang elemen efektif list, yaitu antara indeks pertama (FirstIdx) hingga indeks terakhir (LastIdx). Fungsi ini berguna untuk memverifikasi operasi yang melibatkan elemen-elemen yang telah terisi.

#### 10. boolean Search(List L, ElType X)

Fungsi ini mencari apakah sebuah elemen X terdapat dalam list. Pencarian dilakukan dengan membandingkan semua atribut elemen X terhadap elemen-elemen dalam list. Jika ditemukan elemen yang cocok, fungsi mengembalikan nilai true. Jika tidak, fungsi mengembalikan false.

#### 11. void InsertFirst(List \*L, ElType X)

Fungsi ini menambahkan elemen baru X ke posisi indeks pertama dalam list. Proses ini mengharuskan elemen-elemen yang ada untuk digeser ke kanan terlebih dahulu agar tidak ada elemen yang tertimpa. Fungsi memastikan elemen baru berada di posisi paling awal.

#### 12. void InsertAt(List \*L, ElType X, IdxType i)

Fungsi ini menyisipkan elemen baru **X** pada indeks ke-**i**. Semua elemen mulai dari indeks **i** hingga akhir akan digeser ke kanan untuk memberi ruang bagi elemen baru. Fungsi ini memungkinkan penyisipan data pada posisi tertentu dalam list.

13. void InsertLast(List \*L, ElType X)

Fungsi ini menambahkan elemen baru X di indeks terakhir yang tersedia dalam list. Proses ini memanfaatkan panjang list (Length) untuk menentukan posisi yang tepat bagi elemen baru.

14. void DeleteFirst(List \*L)

Fungsi ini menghapus elemen pada indeks pertama dalam list. Setelah elemen pertama dihapus, elemen-elemen lainnya akan digeser ke kiri untuk mengisi kekosongan. Proses ini memastikan bahwa list tetap terstruktur dengan baik setelah penghapusan.

15. void DeleteAt(List \*L, IdxType i)

Fungsi ini menghapus elemen pada indeks ke-i. Elemen-elemen setelah indeks tersebut akan digeser ke kiri untuk menutup celah yang ditinggalkan oleh elemen yang dihapus. Fungsi ini memberikan fleksibilitas untuk menghapus elemen pada posisi tertentu.

16. void DeleteLast(List \*L)

Fungsi ini menghapus elemen terakhir dalam list. Indeks terakhir diperoleh menggunakan fungsi LastIdx. Elemen terakhir kemudian diubah menjadi Mark untuk menandakan bahwa indeks tersebut sekarang kosong.

17. List Concat(List L1, List L2)

Fungsi ini menggabungkan dua list, L1 dan L2, menjadi sebuah list baru L3. Proses penggabungan dilakukan dengan menyalin elemen dari L1 ke L3, diikuti oleh elemen-elemen dari L2. Jika kapasitas maksimum tercapai sebelum semua elemen tergabung, proses berhenti, dan elemen sisanya tidak dimasukkan.



18. `int myStrcmp(const char *str1, const char *str2)`

Fungsi ini membandingkan dua string karakter per karakter. Jika kedua string memiliki panjang yang sama dan setiap karakter cocok, fungsi mengembalikan nilai 0. Jika terdapat perbedaan, fungsi mengembalikan nilai 1. Fungsi ini digunakan untuk operasi pencarian data berbasis string.

### **3.7 ReadTxt**

ADT ini bertujuan untuk membaca data dari file teks (seperti daftar barang dan pengguna) ke dalam struktur data di program. Fungsi ini sangat penting dalam kesuksesan program ini secara keseluruhan. Berikut adalah fungsi-fungsi yang terdapat pada ReadTxt :

1. `void manualStrcpy(char *dest, const char *source);`

Fungsi ini mengimplementasikan fungsi penyalinan string secara manual. Penyalinan ini dilakukan dari source ke dest karakter demi karakter hingga menemukan karakter null yang menandai akhir string. Jika source adalah "Hello", maka fungsi akan menyalin karakter H, e, l, l, o, dan akhirnya menambahkan '\0' ke dest.

2. `void readtxt(char *filename, ArrayDin *barang, List *user, int *nBarang, int *nUser);`

Fungsi ini bertujuan untuk membaca data dari file teks yang memiliki informasi kemudian menyimpannya ke dalam struktur ArrayDin untuk barang dan List untuk pengguna. Fungsi fopen akan membuka file dengan nama filename dalam mode membaca berupa "r". Jika file gagal dibuka maka program akan mencetak pesan dan keluar dari fungsi. Fungsi ini juga menggunakan konstanta STARTWORDFILE untuk membaca jumlah barang dari baris pertama file. Nilai ini disimpan di nBarang dan diatur oleh Neff pada struktur barang.

### 3.8 Addon

1. `int checkUsernameExists(const char *username, List user, int nUser);\`

Fungsi `checkUsernameExists` bertugas memeriksa apakah sebuah username sudah terdaftar dalam sistem. Fungsi ini menerima tiga parameter: username berupa pointer ke string yang akan diperiksa, user sebagai list berisi data pengguna, dan n-User yang menunjukkan jumlah total pengguna dalam list.

2. `int compareStrings(const char *str1, const char *str2)`

Fungsi `compareStrings` digunakan untuk membandingkan dua string secara karakter per karakter. Fungsi ini menerima dua parameter `str1` dan `str2`, yang masing-masing adalah pointer ke string yang akan dibandingkan. Proses perbandingan dilakukan dengan menggunakan loop yang berjalan hingga salah satu string mencapai karakter null (`'\0'`).

### 3.9 Custom

ADT ini mendefinisikan beberapa tipe data dan konstanta untuk digunakan dalam program C. Terdapat definisi `MAX_LEN` yang membatasi panjang string menjadi 100 karakter, serta tipe data integer yang disamakan dengan `int`. Dua struktur data juga didefinisikan, yaitu `User` yang berisi informasi tentang nama, password, dan jumlah uang, serta `Barang` yang berisi informasi tentang nama barang dan harga.

## 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();
        displayWelcomeMenu();
        printf("\033[1;34mMasukkan perintah Anda: \033[0m");
        STARTWORD();
        Word choice = CurrentWord;

        if (isKataSama(choice, "START")) {
            printf("\n\033[1;34m>> START\033[0m\n");
            start("../save/config.txt", &barang, &user, &nBarang,
&nUser);

            struct timespec req = {2, 0};
            nanosleep(&req, NULL);
            loaded = true;
        }
        else if (isKataSama(choice, "LOAD")) {
            printf("\n\033[1;34m>> LOAD\033[0m ");
            load(&barang, &user, &nBarang, &nUser);

            struct timespec req = {2, 0};
            nanosleep(&req, NULL);
            if (nUser > 0){
                loaded = true;
            }
        }
        else if (isKataSama(choice, "QUIT")) {
            printf("\n\033[1;34m>> QUIT\033[0m\n");
            quit(barang, user, nBarang, nUser, change);
        }
        else if (isKataSama(choice, "HELP")) {
            printf("\n\033[1;34m>> HELP\033[0m\n");

```

```

        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("\033[1;34mMasukkan perintah Anda: \033[0m");
    STARTWORD();
    Word choice = CurrentWord;

    if (isKataSama(choice, "REGISTER")) {
        system("cls || clear");
        displayPurrMart();
        printf("\n\033[1;34m>> REGISTER\033[0m\n");
        addUser(&user, &nUser);
        change = true;

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "LOGIN")) {
        system("cls || clear");
        displayPurrMart();
        printf("\n\033[1;34m>> LOGIN\033[0m\n");
        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("\n\033[1;34m>> QUIT\033[0m\n");
            quit(barang, user, nBarang, nUser, change);
        }
        else {
            printf("\n\033[1;34m>> QUIT\033[0m\n");
            quit(barang, user, nBarang, nUser, change);
        }
    }
    else if (isKataSama(choice, "HELP")) {
        printf("\n\033[1;34m>> HELP\033[0m\n");
        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(user, userIdx);
    printf("\033[1;34mMasukkan perintah Anda: \033[0m");
    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, "WORK")) {
    system("cls || clear");
    displayPurrMart();
    printf("\n\033[1;34m>> 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("\n\033[1;34m>> 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("\n\033[1;34m>> STORE LIST\033[0m\n\n");
    storeList(barang, nBarang);

    struct timespec req = {4, 0};
    nanosleep(&req, NULL);
}
else if (isKataSama(choice, "STORE REQUEST")) {
    printf("\n\033[1;34m>> STORE REQUEST\033[0m\n\n");
    storeRequest(&barang, &antrian);

    struct timespec req = {2, 0};
    nanosleep(&req, NULL);
}
else if (isKataSama(choice, "STORE SUPPLY")) {
    printf("\n\033[1;34m>> 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("\n\033[1;34m>> STORE REMOVE\033[0m\n\n");
    storeremove(&barang, &nBarang);
}

```

```

        change = true;

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "LOGOUT")) {
        printf("\n\033[1;34m>> LOGOUT\033[0m\n\n");
        logoutUser(&userIdx, user);

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
        loggedIn = false;
    }
    else if (isKataSama(choice, "SAVE")) {
        save(barang, user, nBarang, nUser);

        struct timespec req = {2, 0};
        nanosleep(&req, NULL);
    }
    else if (isKataSama(choice, "QUIT")) {
        if (change){
            system("cls || clear");
            displayPurrMart();
            printf("\n\033[1;34m>> QUIT\033[0m\n");
            quit(barang, user, nBarang, nUser, change);
        }
        else {
            printf("\n\033[1;34m>> QUIT\033[0m\n");
            quit(barang, user, nBarang, nUser, change);
        }
    }
    else if (isKataSama(choice, "HELP")) {
        printf("\n\033[1;34m>> HELP\033[0m\n");
        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.

## 6 Data Test

### 6.1 START

Fitur : START

Hasil yang diharapkan :

```
>> START
File konfigurasi aplikasi berhasil dibaca. PURRMART berhasil dijalankan.
*File config.txt berhasil terbaca.
```

Hasil yang didapat :

```
>> START
File konfigurasi aplikasi berhasil dibaca. PURRMART berhasil dijalankan.
```

### 6.2 LOAD

Fitur : LOAD

Hasil yang diharapkan :

```
>> LOAD filename.txt
File filename berhasil dibaca. PURRMART berhasil dijalankan.

>> LOAD filenihil.txt
File filename tidak ditemukan. PURRMART gagal dijalankan.
```

Hasil yang didapat :

```
>> LOAD filename.txt
Save file berhasil dibaca. PURRMART berhasil dijalankan|
```

```
>> LOAD filenihil.txt
Save file tidak ditemukan. PURRMART gagal dijalankan|
```

## 6.3 REGISTER

Fitur : REGISTER

Hasil yang diharapkan :

```
>> REGISTER
```

Username : nama

Password : pass

Akun dengan username nama telah berhasil dibuat. Silakan LOGIN untuk melanjutkan.

```
// Menyimpan nama dan pass ke dalam List user
```

Hasil yang didapat :

```
>> REGISTER
Username: nama
Password: pass
Akun dengan username nama telah berhasil dibuat. Silakan LOGIN untuk melanjutkan.
```

## 6.4 LOGIN

Fitur : LOGIN

Hasil yang diharapkan :

```
>> LOGIN
```

Username : nama

Password : pass

Anda telah login ke PURRMART sebagai nama.

Hasil yang didapat :

```
>> LOGIN
Username: nama
Password: pass
Anda telah login ke PURRMART sebagai nama.
```

## 6.5 LOGOUT

Fitur : LOGOUT

Hasil yang diharapkan :

```
>> LOGOUT

nama telah logout dari sistem PURRMART
Silakan REGISTER/LOGIN kembali untuk melanjutkan
```

Hasil yang didapat :

```
>> LOGOUT

nama telah logout dari sistem PURRMART
Silakan REGISTER/LOGIN kembali untuk melanjutkan
```

## 6.6 WORK

Fitur : WORK

Hasil yang diharapkan :

```
>> WORK

Selamat datang di Work!

Daftar pekerjaan:
1. Evil Lab Assistant (pendapatan = 100, durasi = 14s)
2. OWCA Hiring Manager (pendapatan = 4200, durasi = 21s)
```

3. Cikapundunginator Caretaker (pendapatan = 7000, durasi = 30s)
4. Mewing Specialist (pendapatan = 10000, durasi = 22s)
5. Inator Connoisseur (pendapatan = 997, durasi = 15s)

Masukkan pekerjaan yang dipilih: **Evil Lab Assistant**

Anda sedang bekerja sebagai Evil Lab Asisstant.. harap tunggu.

Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun Anda.

Hasil yang didapat :

```
>> WORK
```

Selamat datang di Work!

Daftar pekerjaan:

1. Evil Lab Assistant (pendapatan = 100, durasi = 14s)
2. OWCA Hiring Manager (pendapatan = 4200, durasi = 21s)
3. Cikapundunginator Caretaker (pendapatan = 7000, durasi = 30s)
4. Mewing Specialist (pendapatan = 10000, durasi = 22s)
5. Inator Connoisseur (pendapatan = 997, durasi = 15s)

Masukkan pekerjaan yang dipilih: Evil Lab Assistant

Anda sedang bekerja sebagai Evil Lab Assistant... harap tunggu.

Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun Anda.

## 6.7 WORK CHALLENGE

Fitur : WORK CHALLENGE

Hasil yang diharapkan :

```
>> WORK CHALLENGE
```

Selamat datang di Work Challenge!

Daftar challenge yang tersedia:

1. Tebak Angka (Biaya: 200 rupiah)
2. W0RDL3 (Biaya: 500 rupiah)

Masukkan challenge yang hendak dimainkan: **W0RDL3**

WELCOME TO W0RDL3, YOU HAVE 6 CHANCES TO ANSWER BEFORE YOU LOSE!

- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)  
Masukkan kata tebakan Anda:

>> WORK CHALLENGE

Selamat datang di Work Challenge!

Daftar challenge yang tersedia:

1. Tebak Angka (Biaya: 200 rupiah)
2. W0RDL3 (Biaya: 500 rupiah)

Masukkan challenge yang hendak dimainkan: **Tebak Angka**  
Tebak angka:

Hasil yang didapat:

>> WORK CHALLENGE

Selamat datang di Work Challenge!

Daftar challenge yang tersedia:

1. Tebak Angka (Biaya: 200 rupiah)
2. W0RDL3 (Biaya: 500 rupiah)

Masukkan challenge yang hendak dimainkan: W0RDL3

WELCOME TO W0RDL3, YOU HAVE 6 CHANCES TO ANSWER BEFORE YOU LOSE!

- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)  
Masukkan kata tebakan Anda: |

```

- - - - -
- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)
Masukkan kata tebakan Anda: BODOH
Hasil:
K% E% R% E% N%
S% M% A* R% T%
B% O% D% O% H*

- - - - -
- - - - -
- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)
Masukkan kata tebakan Anda: FASTE
Hasil:
K% E% R% E% N%
S% M% A* R% T%
B% O% D% O% H*
F% A S% T% E%

- - - - -
- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)
Masukkan kata tebakan Anda: BANDO
Hasil:
K% E% R% E% N%
S% M% A* R% T%
B% O% D% O% H*
F% A S% T% E%
B% A N% D% O%

- - - - -

(Kata bisa dalam bahasa Inggris / Indonesia dan harus kapital)
Masukkan kata tebakan Anda: BASKD
Hasil:
K% E% R% E% N%
S% M% A* R% T%
B% O% D% O% H*
F% A S% T% E%
B% A N% D% O%
B% A S% K% D%

Boo! Anda kalah.
Kata yang benar adalah: HAPPY

```

>> WORK CHALLENGE

Selamat datang di Work Challenge!

Daftar challenge yang tersedia:

1. Tebak Angka (Biaya: 200 rupiah)
2. W0RDL3 (Biaya: 500 rupiah)

Masukkan challenge yang hendak dimainkan: Tebak Angka

Tebak angka: 20

Tebakanmu lebih besar!

Tebak angka: 10

Tebakanmu lebih besar!

Tebak angka: 3

Tebakanmu lebih kecil!

Tebak angka: 6

Tebakanmu lebih kecil!

Tebak angka: 7

Tebakanmu lebih kecil!

Tebak angka: 8

Tebakanmu lebih kecil!

Tebak angka: 9

Tebakanmu benar! +320 rupiah telah ditambahkan ke akun anda.

## 6.8 STORE LIST

Fitur : STORE LIST

Hasil yang diharapkan :

>> STORE LIST

List barang yang ada di toko :

- AK47
- Lalabu
- Ayam Goreng Crisbar

>> STORE LIST

TOKO KOSONG

Hasil yang didapat:

STEI- ITB	<nomor dokumen>	Halaman 31 dari 51 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.		

```
>> STORE LIST
```

```
List barang yang ada di toko :
```

- AK47
- Lalabu
- Ayam Goreng Crisbar

```
>> STORE LIST
```

```
TOKO KOSONG
```

## 6.9 STORE REQUEST

Fitur : STORE REQUEST

Hasil yang diharapkan:

```
>> STORE REQUEST
```

```
Nama barang yang diminta: Bola
```

```
Berhasil menambahkan barang ke antrean
```

```
>> STORE REQUEST
```

```
Nama barang yang diminta: AK47
```

```
Barang dengan nama yang sama sudah ada di toko!
```

```
>> STORE REQUEST
```

```
Nama barang yang diminta: Bola
```

```
Barang dengan nama yang sama sudah ada di antrian!
```

Hasil yang didapat:

```
>> STORE REQUEST
```

```
Nama barang yang diminta: Bola
```

```
Berhasil menambahkan barang ke antrean
```

```
>> STORE REQUEST
```

```
Nama barang yang diminta: AK47
```

```
Barang sudah ada di store
```



```
>> STORE REQUEST
```

```
Nama barang yang diminta: Bola  
Barang sudah ada di antrean
```

## 6.10 STORE SUPPLY

Fitur : STORE SUPPLY

Hasil yang diharapkan :

```
>> STORE SUPPLY  
Apakah kamu ingin menambahkan barang Bola: Terima  
Harga barang: 100  
  
Barang diterima dan dimasukkan ke store.  
  
>> STORE SUPPLY  
Apakah kamu ingin menambahkan barang Bola: Tunda  
  
Barang ditunda dan dimasukkan kembali ke antrian.  
  
>> STORE SUPPLY  
Apakah kamu ingin menambahkan barang Bola: Tolak  
  
Barang ditolak.  
  
>> STORE SUPPLY  
Apakah kamu ingin menambahkan barang Bola: Purry  
  
< Balik ke menu >
```

Hasil yang didapat :

```
>> STORE SUPPLY  
  
Apakah kamu ingin menambahkan barang Bola (Terima/Tunda/Tolak): Terima  
Masukkan harga barang: 100  
Barang diterima dan dimasukkan ke store.
```

```
>> STORE SUPPLY  
  
Apakah kamu ingin menambahkan barang Bola (Terima/Tunda/Tolak): Tunda  
Barang ditunda dan dimasukkan kembali ke antrian.
```

```
>> STORE SUPPLY  
  
Apakah kamu ingin menambahkan barang Bola (Terima/Tunda/Tolak): Tolak  
Barang ditolak
```

```
>> STORE SUPPLY
```

```
Apakah kamu ingin menambahkan barang Bola (Terima/Tunda/Tolak): Purry  
Input tidak dikenali
```

## 6.11 STORE REMOVE

Fitur : STORE REMOVE

Hasil yang diharapkan :

```
>> STORE REMOVE
```

```
Nama barang yang akan dihapus: AK47
```

```
AK47 telah berhasil dihapus.
```

```
>> STORE REMOVE
```

```
Nama barang yang akan dihapus: Inator Neutralizer
```

```
Toko tidak menjual Inator Neutralizer
```

Hasil yang didapat:

```
>> STORE REMOVE
```

```
Nama barang yang akan dihapus: AK47  
AK47 telah berhasil dihapus
```

```
>> STORE REMOVE
```

```
Nama barang yang akan dihapus: Inator Neutralizer  
Toko tidak menjual barang Inator Neutralizer
```

## 6.12 HELP

Fitur : HELP

Hasil yang diharapkan :

```
// Ketika perintah dipanggil pada welcome menu
```

```
>> HELP
```

```
====[ Welcome Menu Help PURRMART]=====
```

1. START -> Untuk masuk sesi baru
2. LOAD -> Untuk memulai sesi berdasarkan file konfigurasi
3. QUIT -> Untuk keluar dari program

```
// Ketika perintah dipanggil pada login menu
>> HELP

=====[ Login Menu Help PURRMART]=====
1. REGISTER -> Untuk melakukan pendaftaran akun baru
2. LOGIN -> Untuk masuk ke dalam akun dan memulai sesi
3. QUIT -> Untuk keluar dari program

// Ketika perintah dipanggil pada main menu
>> HELP

=====[ Menu Help PURRMART]=====
WORK -> Untuk bekerja
WORK CHALLENGE -> Untuk mengerjakan challenge
STORE LIST -> Untuk melihat barang-barang di toko
STORE REQUEST -> Untuk meminta penambahan barang
STORE SUPPLY -> Untuk menambahkan barang dari permintaan
STORE REMOVE -> Untuk menghapus barang
LOGOUT -> Untuk keluar dari sesi
SAVE -> Untuk menyimpan state ke dalam file
QUIT -> Untuk keluar dari program
```

Hasil yang didapat :

```
>> HELP

=====[Welcome Menu Help PURRMART]=====
1. START -> Untuk masuk sesi baru
2. LOAD -> Untuk memulai sesi berdasarkan file konfigurasi
3. QUIT -> Untuk keluar dari program
```

```
>> HELP

=====[Login Menu Help PURRMART]=====
1. REGISTER -> Untuk melakukan pendaftaran akun baru
2. LOGIN -> Untuk masuk ke dalam akun dan memulai sesi
3. QUIT -> Untuk keluar dari program
```

```
>> HELP

====[Main Menu Help PURRMART]====
1. WORK -> Untuk bekerja
2. WORKCHALLENGE -> Untuk mengerjakan challenge
3. STORELIST -> Untuk melihat barang-barang di toko
4. STOREREQUEST -> Untuk meminta penambahan barang
5. STORESUPPLY -> Untuk menambahkan barang dari permintaan
6. STOREREMOVE -> Untuk menghapus barang
7. LOGOUT -> Untuk keluar dari sesi
8. SAVE -> Untuk menyimpan state ke dalam file
9. QUIT -> Untuk keluar dari program
```

## 6.13 SAVE

Fitur : SAVE <filename>.txt

Hasil yang diharapkan :

```
>> SAVE save.txt
```

Save file berhasil disimpan.

```
// File disimpan pada /save/savefile.txt
```

```
>> SAVE config.txt
```

Ini merupakan file konfigurasi, mohon jangan save disini!

```
>> SAVE save
```

Pastikan file disimpan dalam format <nama file>.txt

Hasil yang didapat :

```
>> SAVE save.txt
Save file berhasil disimpan.|
```

```
>> SAVE config.txt
Ini merupakan file konfigurasi, mohon jangan save disini!
```

```
>> SAVE save

Pastikan file disimpan dalam format <nama file>.txt!
```

## 6.14 QUIT

Fitur : QUIT

Hasil yang diharapkan :

```
>> QUIT
Apakah anda ingin save terlebih dahulu(y/n)? N
Save dibatalkan!

----TERIMA KASIH SUDAH BERKUNJUNG DI PURRMART----
```

```
>> QUIT
Apakah anda ingin save terlebih dahulu(y/n)? y
Save (contoh: save.txt): save1.txt
Save berhasil!

----TERIMA KASIH SUDAH BERKUNJUNG DI PURRMART----
```

Hasil yang didapatkan:

```
>> QUIT
Apakah anda ingin save terlebih dahulu(y/n)? N
Save dibatalkan!

-----TERIMA KASIH SUDAH BERKUNJUNG DI PURRMART-----
|
```

```
>> QUIT
Apakah anda ingin save terlebih dahulu(y/n)? y
Save (contoh: save.txt): save1.txt
Save berhasil!

-----TERIMA KASIH SUDAH BERKUNJUNG DI PURRMART-----
```

## 7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1.	START	Untuk mengetahui apakah file konfigurasi terbaca atau tidak	Menulis command “START”	-	File konfigurasi terbaca.	File konfigurasi terbaca.
2.	LOAD	Untuk mengetahui apakah file txt yang di input dapat terbaca atau tidak	Menulis command “LOAD filename.txt”	filename.txt filenihil.txt	Kondisi 1 : File filename.txt terbaca.  Kondisi 2 : File filenihil.txt tidak terbaca	Kondisi 1 : File filename.txt terbaca.  Kondisi 2 : File filenihil.txt tidak terbaca
3.	REGISTER	Untuk mengetahui jika user bisa mendaftarkan akun baru dan tidak mengizinkan user untuk membuat akun dengan username yang sama ketika username sudah terdaftar	Memanggil addUser() tanpa dan dengan username yang sudah ada	admin alstrukdatkere n bagus spakbor	Kondisi 1 : Akun berhasil dibuat. Kondisi 2 : Pesan error username sudah digunakan.	Kondisi 1 : Akun berhasil dibuat. Kondisi 2 : Pesan error username sudah digunakan.

4.	LOGIN	mengecek bila username yang sudah terdaftar bisa login. dan menolak user untuk login jika username tidak terdaftar	Login dengan username yang benar dan salah	admin alstrukdatkern bagus spakbor	Kondisi 1: Login berhasil sebagai testuser Kondisi 2: Pesan error username/pass word salah.	Kondisi 1: Login berhasil sebagai testuser Kondisi 2: Pesan error username/pass word salah.
5.	LOGOUT	logout jika user sudah login.	Memanggil loginUser()	-	Pesan konfirmasi logout.	Pesan konfirmasi logout.
6.	WORK	Memeriksa proses kerja valid	Input “Evil Lab Assistant”	Evil Lab Assistant dan Invalid Job	Kondisi 1: Proses kerja 14s, +100 rupiah. Kondisi 2: Pesan error "Pekerjaan tidak ditemukan!".	Kondisi 1: Proses kerja 14s, +100 rupiah. Kondisi 2: Pesan error "Pekerjaan tidak ditemukan!".
7.	WORK CHALLENGE <Tebak Angka>	Memeriksa game dengan saldo cukup dan kurang	Input angka 1-100 Input dengan saldo kurang	Saldo > biaya game Saldo < biaya game	Kondisi 1: Game berjalan, saldo berkurang. Kondisi 2: Pesan "Uang tidak cukup".	Kondisi 1: Game berjalan, saldo berkurang. Kondisi 2: Pesan "Uang tidak cukup".
8.	WORK CHALLENGE <WORDL3>	Memeriksa validasi input	Input kata 5 huruf dan tidak valid	Kata = 5 huruf Kata < 5 huruf	Kondisi 1: Game berjalan normal. Kondisi 2: Pesan error validasi.	Kondisi 1: Game berjalan normal. Kondisi 2: Pesan error validasi.

8.	STORE LIST	Memeriksa kerja fungsi untuk mencetak list item pada store	Input “STORE LIST” ketika terdapat item dan tidak terdapat item di store.	-	Kondisi 1 : List item tercetak. Kondisi 2 : Pesan “TOKO KOSONG”.	Kondisi 1 : List item tercetak. Kondisi 2 : Pesan “TOKO KOSONG”.
9.	STORE REQUEST	Memeriksa kerja fungsi untuk “meminta” item.	Input nama item yang sudah ada dan yang belum ada pada store serta antrian.	Bola, AK47, Bola (input kedua).	Kondisi 1 : Barang berhasil dimasukkan ke antrean. Kondisi 2 : Pesan “Barang sudah ada di store”. Kondisi 3 : Pesan “Barang sudah ada di antrean”.	Kondisi 1 : Barang berhasil dimasukkan ke antrean. Kondisi 2 : Pesan “Barang sudah ada di store”. Kondisi 3 : Pesan “Barang sudah ada di antrean”.
10.	STORE SUPPLY	Memeriksa kerja fungsi untuk memasukkan item dari antrean ke list.	Input “Terima”, “Tolak”, “Tunda”, serta satu input asal.	Terima, Tolak, Tunda, Purry	Kondisi 1 : Barang berhasil dimasukkan ke dalam list. Kondisi 2 : Barang ditolak dan dihapus dari antrian. Kondisi 3 : Barang ditunda dan dikembalikan ke antrean sebagai urutan terakhir.	Kondisi 1 : Barang berhasil dimasukkan ke dalam list. Kondisi 2 : Barang ditolak dan dihapus dari antrian. Kondisi 3 : Barang ditunda dan dikembalikan ke antrean sebagai urutan terakhir.



					Kondisi 4 : Pesan “Input tidak dikenali” dan kembali ke menu utama.	Kondisi 4 : Pesan “Input tidak dikenali” dan kembali ke menu utama.
11.	STORE REMOVE	Memeriksa kerja fungsi untuk menghapus item pada list store.	Input nama item yang ada dan yang belum ada pada store.	AK47, Batang	Kondisi 1 : Berhasil menghapus item dari list store. Kondisi 2 : Pesan “Toko tidak menjual Batang”.	Kondisi 1 : Berhasil menghapus item dari list store Kondisi 2 : Pesan “Toko tidak menjual Batang”..
12.	HELP	Memeriksa kerja fungsi untuk mencetak daftar command dan kegunaanya	Menulis command “HELP” pada Menu	-	Tercetak daftar command dan kegunaanya pada layar.	Tercetak daftar command dan kegunaanya pada layar.
13.	SAVE	Memeriksa kerja fungsi untuk menyimpan data yang berubah ke dalam sebuah savefile.txt	Menuliskan command “SAVE” kemudian menuliskan nama file save diikuti extension .txt.	save.txt, config.txt, save	Kondisi 1 : Data berhasil tersimpan ke dalam file save.txt.  Kondisi 2: Data tidak berhasil disimpan dan memberi pesan bahwa file input merupakan file	Kondisi 1: Data berhasil tersimpan ke dalam file save.txt.  Kondisi 2: Data tidak berhasil disimpan dan memberi pesan bahwa file input merupakan file

					konfigurasi, input diulang.	konfigurasi, input diulang.
					Kondisi 3: Data tidak berhasil disimpan dan memberi pesan bahwa file input harus txt, input diulang.	Kondisi 3: Data tidak berhasil disimpan dan memberi pesan bahwa file input harus txt, input diulang.
14.	QUIT	Memeriksa kerja fungsi untuk keluar dari program dan mengecek apakah ingin save terlebih dahulu atau tidak	Menulis command “QUIT”, setelah itu input ‘y’ jika ingin save atau ‘n’ jika tidak. Selanjutnya, sama dengan langkah testing save.	y, n	kondisi 1 : Lanjut ke fungsi save dan setelah itu keluar dari program.  kondisi 2 : Langsung keluar dari program	kondisi 1 : Lanjut ke fungsi save dan setelah itu keluar dari program.  kondisi 2 : Langsung keluar dari program

## 8 Pembagian Kerja dalam Kelompok

Ahmad Evander Ruizhi Xavier / 18223064	<ul style="list-style-type: none"> <li>- load</li> <li>- store supply</li> <li>- store remove</li> <li>- save</li> </ul>
Nazwan Siddqi Muttaqin / 18223066	<ul style="list-style-type: none"> <li>- work</li> <li>- work challenge</li> <li>- main</li> </ul>

Keane Putra Lim / 18223056	<ul style="list-style-type: none"> <li>- register</li> <li>- login</li> <li>- logout</li> </ul>
Sebastian Albern Nugroho / 18223074	<ul style="list-style-type: none"> <li>- store list</li> <li>- store request</li> <li>- store supply</li> </ul>
Joan Melkior Silaen / 18223102	<ul style="list-style-type: none"> <li>- start</li> <li>- help</li> <li>- quit</li> </ul>

## 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.

## 1. Command

Pengguna dapat memasukkan *command-command* berikut.

### a. START

START merupakan salah satu *command* yang dimasukkan pertama kali dalam Toko Purrmart. Setelah menekan Enter, dibaca file konfigurasi *default* yang berisi daftar barang pada toko.

### b. LOAD <filename>

LOAD merupakan salah satu *command* yang dimasukkan pertama kali dalam PURRMART. Command ini memiliki satu argumen yaitu *filename* yang merepresentasikan suatu *save file* yang ingin dibuka. *File* didapatkan dari *folder* tertentu, contohnya *save*. Setelah menekan *Enter*, akan dibaca *save file* <filename> yang berisi daftar barang pada toko. Lebih detailnya bisa dilihat pada [Konfigurasi Aplikasi](#).

### c. LOGIN

Login merupakan *command* yang baru dapat dipanggil setelah pengguna memulai sesi. *Login* berguna untuk masuk ke akun di sistem PURRMART yang sudah didaftarkan sebelumnya.

### d. LOGOUT

LOGOUT merupakan salah satu *command* yang baru dapat digunakan setelah pengguna telah memasuki sebuah sesi.

### e. REGISTER

Register merupakan *command* yang baru dapat dipanggil setelah pengguna memulai sesi. *Register* berguna untuk mendaftarkan akun baru ke dalam sistem PURRMART. Sebuah akun setidaknya memiliki atribut *username* dan *password*. **Username dan password hanya terdiri dari 1 kata.**

#### f. WORK

WORK merupakan *command* yang digunakan pengguna untuk mendapatkan uang. Terdapat sejumlah pekerjaan yang bisa dipilih. Setiap pekerjaan memiliki waktu tunggu yang berbeda-beda dan dengan nominal pendapatan yang berbeda-beda pula. Selama pengguna sedang bekerja, maka sistem tidak bisa digunakan hingga pekerjaan selesai dilakukan.

#### g. WORK CHALLENGE

WORK CHALLENGE merupakan *command* alternatif sebagai cara mendapatkan uang dengan melakukan *challenge-challenge* di OWCA. Pemain membutuhkan uang dengan jumlah tertentu untuk bisa memainkan challenge. Uang yang dibayarkan untuk bermain *challenge* tidak akan dikembalikan, meskipun pemain kalah dalam permainan. Terdapat dua *challenge* yang dapat dipilih:

##### a) Tebak Angka

Challenge Tebak Angka merupakan permainan yang meminta pemain menebak sebuah angka yang ditentukan oleh program. Pemain memiliki 10 (sepuluh) kesempatan untuk menebak angka yang benar. Program akan memberikan *feedback* apakah angka tebakan lebih besar, lebih kecil, atau sama dengan angka target. Jumlah kesempatan yang dipakai oleh pengguna akan mempengaruhi uang yang didapatkan.

##### b) WORDL3

Challenge WORDL3 merupakan permainan tebak kata berjumlah lima karakter. Pemain memiliki 6 (enam) kesempatan untuk menebak kata yang benar. Kata harus berupa kata valid, tidak boleh sekadar *string* acak, bahasa dibebaskan (disarankan bahasa Indonesia/Inggris). Pada setiap giliran, program akan mencetak ulang kata yang dimasukan, tetapi dengan penanda tertentu. Huruf yang benar dan berada pada tempat yang tepat dicetak biasa. Huruf yang benar, tetapi berada di tempat yang salah diberi tanda “\*” setelah hurufnya. Huruf yang tidak ada sama sekali pada kata diberi tanda “%” setelah hurufnya.

#### h. STORE LIST

STORE LIST adalah *command* yang digunakan untuk melihat barang-barang apa saja yang ada di dalam toko. **Setiap barang yang ditampilkan haruslah bersifat *unique*.**

#### i. STORE REQUEST

STORE REQUEST adalah *command* yang digunakan untuk meminta penambahan barang baru ke dalam toko. Barang-barang yang diminta akan disimpan di dalam sebuah antrian dan akan dimasukkan ke toko menggunakan *command* selanjutnya. **Nama barang yang masuk tidak boleh sama dengan nama barang yang sudah ada di toko atau di antrian.**

**j. STORE SUPPLY**

STORE SUPPLY adalah *command* yang digunakan untuk menambahkan barang baru ke dalam toko berdasarkan antrian permintaan. Barang yang berada pada antrian paling depan akan dimasukan ke toko. Pengguna dapat menerima, menunda, atau menolak permintaan.

- Jika diterima, maka program akan meminta harga dari barang dan dimasukan ke toko.
- Jika ditunda, maka barang akan kembali masuk ke antrian
- Jika ditolak, maka barang akan dihapus dari antrian

Harus terdapat validasi agar harga barang merupakan angka yang valid (berupa angka dan bernilai lebih dari nol).

**k. STORE REMOVE**

STORE REMOVE adalah *command* yang dapat menghapus barang yang ada di toko. Akan dilakukan *input* akan barang yang akan dihapus. Beri tahu apabila proses berhasil (barang terdapat pada toko dan berhasil dihapus) ataupun tidak (barang tidak terdapat di toko).

**l. HELP**

HELP merupakan *command* yang digunakan menampilkan daftar *command* yang mungkin untuk dieksekusi dengan deskripsinya. Penjelasan dari deskripsi dibebaskan selama masih mendeskripsikan *command* sesuai dengan spek.

**m. SAVE <filename>**

SAVE merupakan *command* yang digunakan untuk menyimpan *state* aplikasi terbaru ke dalam suatu *file*. Command SAVE memiliki satu argumen yang merepresentasikan nama *file* yang akan disimpan. Penyimpanan dilakukan pada *folder* tertentu, misal *folder save*.

**n. QUIT**



QUIT merupakan *command* yang digunakan untuk keluar dari sesi aplikasi PURRMART.

## 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

<b>Tanggal : 24 November 2024</b>	<b>Catatan Asistensi:</b>  Asistensi pertama ini membahas mengenai fungsi-fungsi yang masih belum jelas, penulisan laporan, serta hal-hal teknis lainnya. Setelah melakukan asistensi, didapatkan hasil sebagai berikut : <ol style="list-style-type: none"><li>1. Untuk pengerjaan teknis dibebaskan, yang terpenting jangan sampai program crash saat demo.</li><li>2. Pengerjaan milestone 2 akan melanjutkan pengerjaan dari milestone 1, tidak membuat program baru. Laporan juga akan dikerjakan melanjutkan dari milestone 1.</li><li>3. Log Activity berisikan tanggal, lokasi, NIM mahasiswa yang terlibat dalam aktivitas, serta aktivitas yang dilakukan.</li><li>4. Untuk milestone 1, kemungkinan tidak akan ada pengunduran deadline.</li></ol>
<b>Tempat : Zoom Meeting</b>	
<b>Kehadiran Anggota Kelompok:</b>  1. Ahmad Evander Ruizhi Xavier (18223064)    2. Nazwan Siddqi Muttaqin (18223066)    3. Sebastian Albern Nugroho (18223074)	

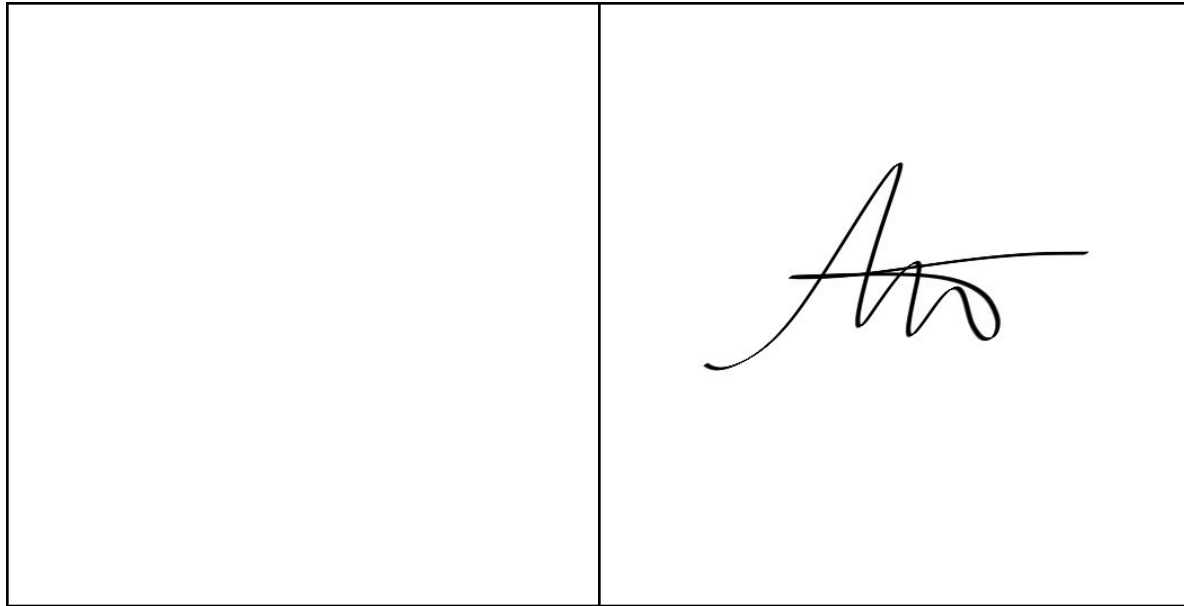


4. Joan Melkior Silaen (18223102)

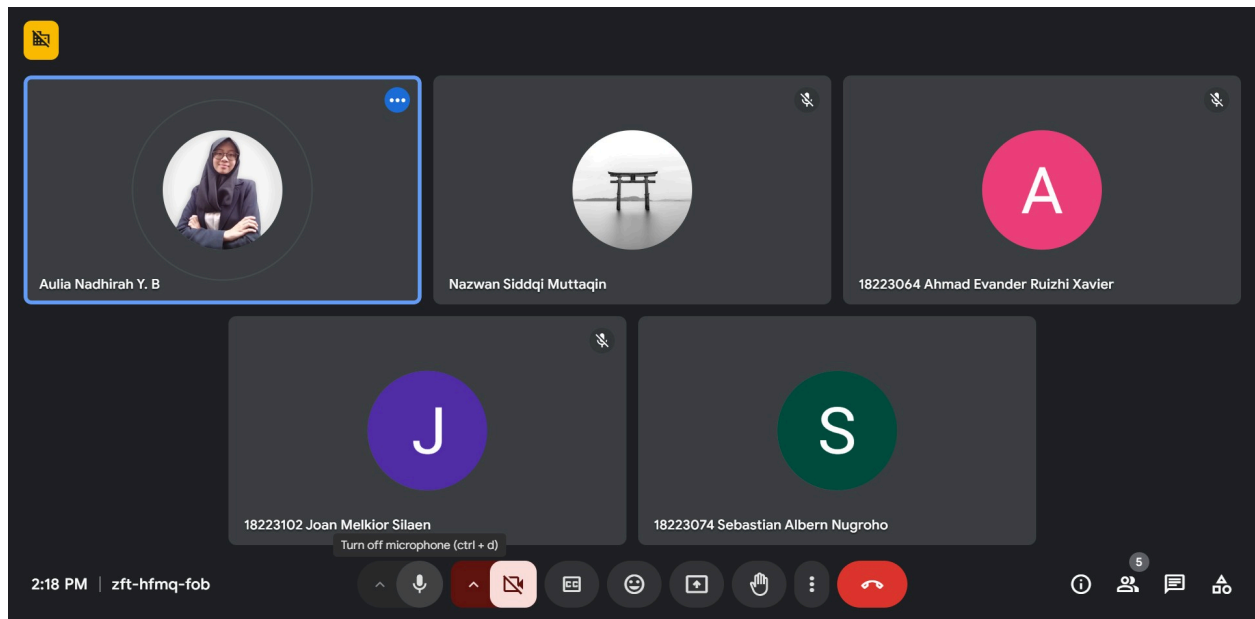


Tanda Tangan Asisten:





Dokumentasi:



### 9.3 Log Activity Anggota Kelompok

Tanggal	NIM	Lokasi	Aktivitas
14/11/2024	18223064	R. Multimedia	Pembagian tugas awal

STEI- ITB	<nomor dokumen>	Halaman 49 dari 51 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.		

	18223066 18223056 18223074 18223102	Labtek V	
18/11/2024	18223066	Rumah	Memulai pengerjaan work dan work_challenge
18/11/2024	18223074	Kost	Memulai pengerjaan storeList, storeRequest, dan storeSupply
19/11/2024	18223102	Kost	Memulai pengerjaan start, quitm dan help
19/11/2024	18223066	Rumah	Menyelesaikan work dan work_challenge
21/11/2024	18223064 18223066 18223056 18223074 18223102	Labtek V	Kerja kelompok update hasil kerja masing masing
22/11/2024	18223064 18223066 18223056 18223074 18223102	Labtek V	Mulai melakukan penyelarasan tiap fungsi
22/11/2024	18223074	Kost	Commit dan push storeList
23/11/2024	18223064	Kost	Menyelesaikan save, load, dan storeRemove
23/11/2024	18223064 18223066 18223056 18223074 18223102	Kost	Memulai pengerjaan laporan
24/11/2024	18223064	Kost	Membantu pengerjaan storeSupply, menyesuaikan fungsi
24/11/2024	18223064	Zoom Meeting	Asistensi 1

	18223066 18223056 18223074 18223102		
24/11/2024	18223064 18223074	Kost	Commit dan push storeRequest dan storeSupply
24/11/2024	18223066	Rumah	Menggabungkan semua fungsi ke dalam main dan membuat tampilannya lebih menarik
24/11/2024	18223064 18223066 18223056 18223074 18223102	Kost	Proses debugging
25/11/2024	18223064 18223066 18223056 18223074 18223102	Labtek V dan kost	Pengumpulan milestone 1 tugas besar Algoritma dan Struktur Data STI