

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data STI

PURRMART


Dipersiapkan oleh:

Kelompok 05

Adam Joaquin Girsang	/18223089
Timothy Marvine	/18223021
A. Nurul Aqeela Amin	/18223019
Muhammad Naufal Fathan	/18223059
Khairunnisa Hurun 'lin	/18221004

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-01-05</i>		<i>40</i>
		<i>Revisi</i>	<i><no revisi></i>	<i><Tgl release></i>

Daftar Isi

1	Ringkasan	3
2	Struktur Data (ADT)	5
2.1	ADT Array	5
2.1.1.	Sketsa Struktur	5
2.1.2.	Persoalan yang diselesaikan	7
2.1.3	Alasan Pemilihan	7
2.1.4	Implementasi	7
2.2	ADT Mesin Kata	7
2.2.1	Sketsa Struktur Data	7
2.2.2	Persoalan yang diselesaikan	10
2.2.3	Alasan pemilihan	10
2.2.4	Implementasi	10
2.3	ADT Mesin Karakter	10
2.3.1	Sketsa Struktur	10
2.3.2	Persoalan yang diselesaikan	11
2.3.3	Alasan pemilihan	11
2.3.4	Implementasi	11
2.4	ADT Queue	11
2.4.1	Sketsa Struktur	11
2.4.2	Persoalan yang diselesaikan	13
2.4.3	Alasan pemilihan	13
2.4.4	Implementasi	13
2.5	ADT List	13
2.5.1	Sketsa Struktur	13
2.5.2	Persoalan yang diselesaikan	14
2.5.3	Alasan pemilihan	14
2.5.4	Implementasi	14
3	Program Utama	15
3.1	Main Menu	15
3.2	Pembacaan dan Inisialisasi File Konfigurasi	15
3.3	Pemanggilan Command	16
4	Algoritma-Algoritma Menarik	17
4.1	Prosedur STORE LIST	17
4.2	Prosedur STORE REMOVE	18
5	Data Test	19

5.1 Data Test MAIN MENU	19
5.2 Data Test START	19
5.3 Data Test LOAD	20
5.4 Data Test LOGIN	20
5.5 Data Test LOGOUT	22
5.6 Data Test REGISTER	22
5.7 Data Test WORK	23
5.8 Data Test WORK CHALLENGE	23
5.9 Data Test STORE LIST	27
5.10 Data Test STORE REQUEST	27
5.11 Data Test STORE SUPPLY	28
5.12 Data Test STORE REMOVE	29
5.13 Data Test HELP	30
5.14 Data Test SAVE	31
5.15 Data Test QUIT	31
6 Test Script	32
7 Pembagian Kerja dalam Kelompok	35
8 Lampiran	38
8.1 Deskripsi Tugas Besar 1	38
8.2 Notulen Rapat	38
8.3 Log Activity Anggota Kelompok	39

1 Ringkasan

Tugas Besar 1 IF2111 meminta mahasiswa untuk mengembangkan PURRMART, sebuah aplikasi yang menggunakan bahasa C untuk simulasi *Command Line Interface* (CLI) yang berfungsi sebagai sistem jual-beli virtual. Dalam simulasi ini, PURRMART bertindak sebagai "Toko Borma," sebuah pemasok senjata perang untuk Agen Purry, karakter fiksi yang membutuhkan senjata untuk menyelesaikan misi rahasia tanpa perlu bertemu langsung dengan pemasok. PURRMART menyediakan berbagai fitur, termasuk manajemen akun, simulasi kerja, tantangan interaktif, serta pengelolaan toko. Pengguna dapat mendaftarkan akun baru menggunakan command REGISTER, masuk dengan LOGIN, dan keluar dengan LOGOUT. Selain itu, PURRMART memungkinkan pengguna untuk bekerja melalui command WORK, memilih dari daftar pekerjaan dengan durasi dan penghasilan tertentu, atau menyelesaikan tantangan melalui WORK CHALLENGE, seperti Tebak Angka atau WORDL399, untuk mendapatkan hadiah tambahan.

Dalam hal pengelolaan toko, PURRMART menyediakan fitur seperti STORE LIST untuk melihat barang-barang yang tersedia, STORE REQUEST untuk menambahkan permintaan barang baru ke antrian, dan STORE SUPPLY untuk memproses antrian barang, baik dengan menerima, menunda, atau menolak barang yang diminta. Pengguna juga dapat menghapus barang dengan STORE REMOVE. File konfigurasi atau penyimpanan dapat dikelola melalui command START untuk memuat file default atau LOAD untuk file tertentu, serta SAVE untuk menyimpan status aplikasi saat ini. Aplikasi ini dirancang agar memberikan pengalaman interaktif melalui berbagai pesan dan validasi yang memastikan penggunaan fitur yang tepat. Secara keseluruhan, PURRMART menghadirkan pengalaman simulasi jual beli yang komprehensif dengan tantangan yang menarik dan pengelolaan data yang terorganisir.

Laporan ini berisi penjelasan tentang program yang telah kami buat. Bagian pertama berisi ringkasan berisi deskripsi umum persoalan dengan singkat. Bagian kedua berisi tentang penjelasan tambahan spesifikasi tugas. Bagian ketiga tentang struktur data yang digunakan agar

program dapat berjalan. Bagian keempat berisi tentang program utama. Bagian kelima berisi tentang algoritma-algoritma menarik yang digunakan di dalam program ini. Bagian keenam berisi tentang *data test*. Bagian ketujuh berisi tentang *test script* yang berisi skenario test untuk semua fitur. Bagian kedelapan berisi tentang pembagian kerja kelompok. Bagian kesembilan berisi tentang lampiran yang terkait dengan dokumen.

Program ini digunakan menggunakan bahas C dengan struktur data yang telah dipelajari pada mata kuliah IF2111 Algoritma dan Struktur Data. ADT-ADT yang digunakan yaitu array, mesin kata, mesin karakter, queue, dan list.

2 Struktur Data (ADT)

Pada pembuatan aplikasi PURRMART ini, kami menggunakan beberapa struktur data untuk menyelesaikan persoalan-persoalan pada Tugas Besar, yaitu ADT Array , ADT Mesin Kata, ADT Mesin Karakter, ADT Queue, dan ADT List.

2.1 ADT Array

2.1.1. Sketsa Struktur

```
#ifndef __ARRAY_DINAMIK__
#define __ARRAY_DINAMIK__

#include"queue.h"

#define InitialSize 10

typedef struct {
    char name[50];
    int price;
} Barang;

typedef struct {
    Barang *A;
    int Capacity;
    int Neff;
} ArrayDin;

ArrayDin MakeArrayDin();
/* I.S. : Tidak ada array dinamik yang teralokasi
   F.S. : Sebuah array dinamik baru dialokasikan dengan kapasitas awal sebesar
```

```

InitialSize dan Neff = 0 */

void DeallocateArrayDin(ArrayDin *array);
/* I.S. : Array menunjuk ke sebuah array dinamik yang sudah dialokasikan
   F.S. : Memori untuk array A telah dibebaskan, Capacity menjadi 0, dan Neff
   menjadi 0 */

boolean IsEmptyDin(ArrayDin array);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Fungsi mengembalikan true jika Neff == 0 */

int LengthDin(ArrayDin array);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Fungsi mengembalikan jumlah elemen efektif(Neff) dari array*/

char *Get(ArrayDin array, IdxType i);
/* I.S. : Sebuah array dinamik dengan elemen yang terdefinisi
   F.S. : Fungsi mengembalikan pointer ke elemen array pada indeks i */

int GetCapacity(ArrayDin array);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Fungsi mengembalikan kapasitas maksimum dari array */

void InsertAt(ArrayDin *array, char *el, IdxType i, int x);
/* I.S. : Array mungkin penuh atau memiliki ruang kosong
   F.S. : Jika kapasitas penuh, array akan direalokasi dengan kapasitas lebih besar
   */

void InsertLast(ArrayDin *array, Barang el);
/* I.S. : Array mungkin penuh atau memiliki ruang kosong
   F.S. : Jika kapasitas penuh, array akan direalokasi dengan kapasitas lebih
   besar */

void InsertFirst(ArrayDin *array, Barang el);
/* I.S. : Array mungkin penuh atau memiliki ruang kosong
   F.S. : Jika kapasitas penuh, array akan direalokasi dengan kapasitas lebih
   besar */

void DeleteAt(ArrayDin *array, IdxType i);
/* I.S. : Array tidak kosong
   F.S. : Elemen pada indeks i dihapus, Elemen setelah indeks i bergeser ke kiri,
   dan Neff berkurang 1 */

void DeleteLast(ArrayDin *array);
/* I.S. : Array tidak kosong
   F.S. : Elemen terakhir dihapus */

void DeleteFirst(ArrayDin *array);
/* I.S. : Array tidak kosong
   F.S. : Elemen pertama dihapus, Semua elemen bergeser ke kiri */

```

```

void PrintArrayDin(ArrayDin array);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Elemen array dicetak ke layar */

ArrayDin CopyArrayDin(ArrayDin array);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Mengembalikan array dinamik baru yang merupakan salinan dari array asli
*/

boolean SearchArrayDin(ArrayDin array, char *el);
/* I.S. : Sebuah array dinamik yang mungkin kosong atau berisi elemen
   F.S. : Mengembalikan true jika elemen el ditemukan di array */

#endif

```

2.1.2. Persoalan yang diselesaikan

ADT Array digunakan untuk menampilkan daftar barang yang ada di dalam program

2.1.3 Alasan Pemilihan

Kami menggunakan ADT Array Dinamis karena jumlah permintaan barang pada fungsi store request dan store supply bervariasi jumlahnya, oleh karena itu memori yang dialokasikan hanya sesuai jumlah barang

2.1.4 Implementasi

Implementasi array dalam program ini ada di dalam kode arraydin.c dalam folder ADT (src/ADT/arraydin.c)

2.2 ADT Mesin Kata

2.2.1 Sketsa Struktur Data

```

/* File : mesinkata.h */

/* Definisi Mesin Kata: Model Akuisisi Versi I */

#ifndef __MESINKATA_H__
#define __MESINKATA_H__

#include "boolean.h"
#include "mesinkarakter.h"

#define NMax 50
#define BLANK ' '
#define NEWLINE '\n'

```

```

typedef struct
{
    char TabWord[NMax]; /* container penyimpan kata, indeks yang dipakai
[0..NMax-1] */
    int Length;
} Word;

/* State Mesin Kata */
extern boolean EndWord;
extern Word currentWord;

void IgnoreBlanks();
/* Mengabaikan satu atau beberapa BLANK dan/atau ENTER (NEWLINE)
    I.S. : currentChar sembarang
    F.S. : currentChar ≠ BLANK atau currentChar = MARK */
// void IgnoreBlanks2();
/* Mengabaikan beberapa BLANK */

void STARTWORD();
/* Kata yang dibaca dengan prosedur START() yang akan membaca dari input user,
akuisisi kata menggunakan CopyWord.
    I.S. : currentChar sembarang
    F.S. : EndWord = true, dan currentChar = MARK;
           atau EndWord = false, currentWord adalah kata yang sudah
diakuisisi, currentChar karakter pertama sesudah karakter terakhir kata */

void STARTWORDFILE(char *FileName);
/* Mengakses pita dengan satu word adalah satu kalimat (mengabaikan BLANK)
    I.S. : currentChar sembarang
    F.S. : EndWord = true, dan currentChar = MARK;
           atau EndWord = false, currentWord adalah kata yang sudah diakuisisi dari
file,
           currentChar karakter pertama sesudah karakter terakhir kata */

void STARTSENTENCE();
/* Mengakses pita dengan satu word adalah satu kata yang sebenarnya */

void STARTENTER();
/* Mesin menerima inputan enter, state program berganti */

void ADVWORD();
/* I.S. : currentChar adalah karakter pertama kata yang akan diakuisisi
    F.S. : currentWord adalah kata terakhir yang sudah diakuisisi,
           currentChar adalah karakter pertama dari kata berikutnya, mungkin MARK
           Jika currentChar = MARK, EndWord = true.
    Proses : Akuisisi kata menggunakan procedure SalinWord */

void ADVSENTENCE();

```



```

/* Akuisisi kalimat menggunakan prosedur CopySentence */

void CopyWord();
/* Mengakuisisi kata, kata merupakan seluruh input (bisa merupakan kalimat),
menyimpan dalam currentWord
    I.S. : currentChar adalah karakter pertama dari kata
    F.S. : currentWord berisi kata yang sudah diakuisisi;
           currentChar = BLANK atau currentChar = MARK;
           currentChar adalah karakter sesudah karakter terakhir yang diakuisisi.
           Jika panjang kata melebihi NMax, maka sisa kata "dipotong" */

void CopySentence();
/* Mengakuisisi kata, menyimpan dalam currentWord
    I.S. : currentChar adalah karakter pertama dari kata
    F.S. : currentWord berisi kata yang sudah diakuisisi;
           currentChar = BLANK atau currentChar = MARK;
           currentChar adalah karakter sesudah karakter terakhir yang
diakuisisi. Jika panjang kata melebihi NMax, maka sisa kata "dipotong" */

int Strlen(char *s);
/* I.S. : Proses belum dimulai
    F.S. : Mengembalikan panjang string s */

void WordToString(Word Kata, char *s);
/* Proses : Menerima kata dalam bentuk Word lalu mengubahnya ke bentuk string
    I.S. : Word terdefinisi
    F.S. : terbentuk sebuah string s yang berisi char dari currentWord */

int stringToInteger(char *str);
/* I.S. : String terdefinisi
    F.S. : Menghasilkan integer yang merupakan hasil konversi dari string */

boolean WordCompareString(Word Kata, char *s);
/* Proses : Membandingkan kata dengan string, menghasilkan true jika sama
    I.S. : Word terdefinisi, string juga terdefinisi
    F.S. : menghasilkan true jika kata sama dengan ripresentasinya pada string
input */

boolean WordCompare(Word input1, Word input2);
/* Proses : Membandingkan kata dengan kata, menghasilkan true jika kata sama
    I.S. : Word terdefinisi
    F.S. : menghasilkan true jika kedua kata sama, false jika tidak */

boolean WordCompareKapital(Word input1, Word input2);
/* Proses : Membandingkan kata dengan kata, menghasilkan true jika kata sama,
kata yg sama adalah tidak dibedakan kapital dan tidaknya
    I.S. : Word terdefinisi
    F.S. : menghasilkan true jika kedua kata sama, false jika tidak */

void PrintKata(Word Kata);

```

```

/* Proses : Menuliskan tipe bentukan kata ke layar
   I.S. : Word terdefinisi
   F.S. : kata yang disimpan dalam Word Kata tertulis di layar */

#endif

```

2.2.2 Persoalan yang diselesaikan

ADT Mesin Kata digunakan sebagai pengganti scanf untuk membaca karakter atau kata pada program.

2.2.3 Alasan pemilihan

Alasan digunakannya ADT Mesin Kata karena mempermudah program dalam membaca sebuah karakter atau kata dan digunakan untuk menerima input dan command dari user.

2.2.4 Implementasi

ADT Mesin Kata diimplementasikan dalam 'src/ADT/mesinkata.c'.

2.3 ADT Mesin Karakter

2.3.1 Sketsa Struktur

```

#ifndef __MESIN_KAR_H_
#define __MESIN_KAR_H_
#include "boolean.h"
#define MARK ';'
/* State Mesin */
extern FILE *pita;
extern char currentChar;
extern boolean EOP;

void START();
/* Mesin siap dioperasikan. Pita merupakan stdin yang adalah inputan dari
pengguna.
   Karakter pertama yang ada pada pita posisinya adalah pada jendela.
   Pita baca diambil dari stdin.
   I.S. : sembarang
   F.S. : currentChar adalah karakter pertama pada pita
         Jika currentChar != MARK maka EOP akan padam (false)
         Jika currentChar = MARK maka EOP akan menyala (true) */

void STARTFILE(char *FileName);
/* Mesin siap dioperasikan. Pita merupakan file yang diakses dan disiapkan untuk
dibaca.
   Karakter pertama yang ada pada pita posisinya adalah pada jendela.

```

```

Pita baca diambil dari stdin.
I.S. : sembarang
F.S. : currentChar adalah karakter pertama pada pita
      Jika currentChar != MARK maka EOP akan padam (false)
      Jika currentChar = MARK maka EOP akan menyala (true) */

void ADV();
/* Pita dimajukan satu karakter.
   I.S. : Karakter pada jendela = currentChar, currentChar != MARK
   F.S. : currentChar adalah karakter berikutnya dari currentChar yang lama,
         currentChar mungkin = MARK
         Jika currentChar = MARK maka EOP akan menyala (true) */

// void ADV2();
/* Prosedur ADV yang digunakan pada ADVWORD2 */

char GetCC();
/* Mengirimkan currentChar */

boolean IsEOP();
/* Mengirimkan true jika currentChar = MARK */

#endif

```

2.3.2 Persoalan yang diselesaikan

ADT Mesin Karakter digunakan sebagai pengganti dari scanf, serta sebagai persyaratan dari ADT Mesin Kata.

2.3.3 Alasan pemilihan

ADT Mesin Karakter dapat membantu program dalam membaca karakter, menggeser atau memajukan karakter, memasukkan karakter, dan juga dibutuhkan untuk ADT Mesin Kata.

2.3.4 Implementasi

ADT Mesin Karakter diimplementasikan dalam 'src/ADT/mesinkarakter.c'.

2.4 ADT Queue

2.4.1 Sketsa Struktur

```

/* File : queue.h */
/* Definisi ADT Queue dengan representasi array secara eksplisit dan alokasi
statik */

```

```

#ifndef QUEUE_H
#define QUEUE_H

#include "..\boolean.h"

#define IDX_MAX 99
#define IDX_UNDEF -1

/* Definisi elemen dan address */
typedef char Produk[50];
typedef int IdxType;

typedef struct {
    Produk Tab[IDX_MAX + 1];
    IdxType idxHead;
    IdxType idxTail;
} Queue;

/* ***** AKSES (Selektor) ***** */
#define IDX_HEAD(q) (q).idxHead
#define IDX_TAIL(q) (q).idxTail
#define HEAD(q) (q).Tab[(q).idxHead]
#define TAIL(q) (q).Tab[(q).idxTail]

/* *** Kreator *** */
void CreateQueue (Queue * Q);
/* I.S. : sembarang
   F.S. : sebuah q kosong terbentuk dengan kondisi seperti ini: */
/* - Index head bernilai IDX_UNDEF */
/* - Index tail bernilai IDX_UNDEF */

/* ***** Prototype ***** */
boolean IsEmpty (Queue Q);
/* Mengirim true jika q kosong */
boolean IsFull (Queue Q);
/* Mengirim true jika tabel penampung elemen q sudah penuh */
/* yaitu IDX_TAIL akan selalu di belakang IDX_HEAD dalam buffer melingkar */

int Length (Queue Q);
/* Mengirimkan banyaknya elemen queue. Mengirimkan 0 jika q kosong. */

void enqueue (Queue * Q, Produk X);
/* I.S. : q mungkin kosong, tabel penampung elemen q TIDAK penuh */
/* F.S. : val menjadi TAIL yang baru, IDX_TAIL "mundur" dalam buffer melingkar. */

void dequeue (Queue * Q);
/* I.S. : q tidak mungkin kosong */
/* F.S. : val = nilai elemen HEAD pd I.S., IDX_HEAD "mundur"; q mungkin kosong */

boolean IsMemberQ(Queue Q, Produk x);

```

```

/* *** Display Queue *** */
void DisplayQueue(Queue q);
/* Proses : Menuliskan isi Queue dengan traversal, Queue ditulis di antara kurung
siku; antara dua elemen dipisahkan separator "koma", tanpa tambahan karakter di
depan, di tengah, atau di belakang, termasuk spasi dan enter */
/* I.S. : q boleh kosong */
/* F.S. : Jika q tidak kosong: [e1,e2,...,en] */
/* Contoh : jika ada tiga elemen bernilai 1,15,25 akan dicetak: [1,15,25]
/* Jika Queue kosong : menulis [] */

boolean strCmpr(char *a, char *b);
/* I.S. : Fungsi menerima dua parameter pointer ke karakter (char *a dan char *b)
yang merujuk pada awal dua string yang akan dibandingkan.
F.S. : Fungsi akan mengembalikan nilai true jika kedua string sama persis
(termasuk panjangnya).*/

void strCopy(Produk a, Produk b);
/* I.S. : Fungsi menerima dua parameter: Produk dest (string tujuan) dan Produk
src (string sumber).
F.S. : String di src disalin ke dest karakter demi karakter hingga mencapai \0.
*/

#endif

```

2.4.2 Persoalan yang diselesaikan

Persoalan yang diselesaikan dengan ADT Queue adalah mengantri permintaan penambahan barang dalam sistem.

2.4.3 Alasan pemilihan

ADT Queue dipilih karena digunakan untuk merepresentasikan urutan permintaan penambahan barang.

2.4.4 Implementasi

Implementasi ADT Queue terdapat dalam 'src/ADT/queue.c'

2.5 ADT List

2.5.1 Sketsa Struktur

```

#ifndef ADTLIST_H
#define ADTLIST_H

#include "storereq.h"

```

```

typedef struct {
    Barang Item[20];
} Daftar;

void CreateList(Daftar *l);
/* I.S. sembarang */
/* F.S. Terbentuk List kosong dengan Neff bernilai 0 */

boolean IsEmptyList(Daftar l);
/* Mengirimkan true jika Daftar l kosong (Neff = 0), mengirimkan false jika tidak */

boolean IsFullList(Daftar l);
/* Mengirimkan true jika daftar l penuh (Neff == MAX_ ITEMS), false jika tidak */

void InsertList(Daftar *l, Barang b);
/* I.S. l terdefinisi, mungkin kosong. b terdefinisi */
/* F.S. b ditambahkan ke akhir daftar jika masih ada ruang. Jika daftar penuh,
elemen tidak ditambahkan. */
void DeleteList(Daftar *l, int idx);
/* I.S. l terdefinisi, tidak kosong. idx adalah indeks valid dalam daftar (0 ≤ idx < Neff). */
/* F.S. Elemen pada indeks idx dihapus, elemen setelahnya digeser ke depan */

void DisplayList(Daftar l);
/* I.S. l terdefinisi, mungkin kosong. */
/* F.S. Elemen daftar ditampilkan satu per satu. jika kosong, tampilkan pesan
bahwa daftar kosong. */

#endif

```

2.5.2 Persoalan yang diselesaikan

Persoalan yang diselesaikan dengan ADT List adalah mengetahui jumlah item yang ada di store menggunakan akses Neff.

2.5.3 Alasan pemilihan

ADT list dipilih karena dapat menyimpan data username dan data item yang terdapat pada store, serta relatif mudah digunakan.

2.5.4 Implementasi

Implementasi ADT List terdapat dalam 'src/ADT/arraydin.h'

3 Program Utama

3.1 Main Menu

Program utama dinamai “main.c”, file ini mencakup semua file ADT yang telah dibuat. Program dimulai dengan menampilkan tampilan pilihan menu utama (*main menu*) yang berisikan pilihan *command* seperti START untuk memulai progres yang baru, LOAD untuk melanjutkan progres yang telah ada, dan HELP untuk melihat cara menggunakan program. Pengguna dapat mengelola akun dengan REGISTER, LOGIN, dan LOGOUT, serta mendapatkan penghasilan melalui WORK atau tantangan seperti Tebak Angka dan WORDL3 melalui WORK CHALLENGE. Pengelolaan toko mencakup fitur STORE LIST untuk melihat barang, STORE REQUEST untuk menambahkan permintaan barang, STORE SUPPLY untuk memproses antrian permintaan, dan STORE REMOVE untuk menghapus barang. Progres aplikasi dapat disimpan menggunakan SAVE. Program ini akan berhenti looping jika user memberikan input command berupa QUIT.

3.2 Pembacaan dan Inisialisasi File Konfigurasi

Dalam pembacaan, perlu diketahui format file konfigurasi, spesifikasinya adalah sebagai berikut:

1. Barisan pertama adalah bilangan bulat positif N yang menunjukkan berapa banyak barang di dalam sistem
2. Selanjutnya, sejumlah N baris menyatakan nama barang beserta harganya dengan format <Harga barang> <Nama barang>
3. Baris selanjutnya adalah bilangan bulat positif M yang menunjukkan berapa banyak pengguna di dalam sistem
4. Selanjutnya, sejumlah M baris menyatakan nama pengguna, password, jumlah uang dengan format <Uang> <Nama> <Password>

Inisialisasi penamaan file dimulai dengan menentukan path penyimpanan file, kemudian file dibaca dengan fungsi STARTWORDFILE, apabila fungsi mengakses EndWord, maka file

tidak ditemukan atau kosong. Apabila file ditemukan, akan dibaca isi dari file yang merupakan <Harga barang> <Nama barang> sejumlah N dan <Uang> <Nama> <Password> sejumlah M.

3.3 Pemanggilan Command

3.3.1 START

START merupakan fungsi yang pertama kali dijalankan untuk memproses program. START akan membaca file konfigurasi default yang berisi list barang dan list user.

3.3.2 LOAD <filename>

LOAD merupakan fungsi yang digunakan untuk membaca suatu save file yang ingin dibuka.

3.3.3 LOGIN

LOGIN merupakan fungsi yang digunakan agar user dapat masuk ke akunnya dan menjalankan programnya.

3.3.4 LOGOUT

LOGOUT merupakan fungsi yang digunakan agar user dapat keluar dari akunnya

3.3.5 REGISTER

REGISTER merupakan fungsi yang digunakan agar user dapat mendaftarkan akunnya pada database.

3.3.6 WORK

WORK merupakan fungsi yang digunakan untuk user dapat mendapatkan uang dengan bekerja menurut perannya masing-masing, Setiap peran memiliki pendapatan dan durasi bekerja yang berbeda-beda.

3.3.7 WORK CHALLENGE

WORK CHALLENGE merupakan fungsi yang digunakan untuk user dapat menggunakan uangnya yang didapatkan melalui WORK untuk memainkan 2 challenge yaitu Tebak Angka dan juga WORDLE399 . Dari challenge tersebut, apabila user berhasil menamatkan masing-masing challenge akan mendapatkan uang sebagai hadiah.

3.3.8 STORE LIST

STORE LIST merupakan fungsi yang digunakan user dapat melihat barang-barang apa saja yang ada di toko.

3.3.9 STORE REQUEST

STORE REQUEST merupakan fungsi yang digunakan untuk user dapat meminta penambahan barang di toko.

3.3.10 STORE SUPPLY

STORE SUPPLY merupakan fungsi yang digunakan untuk user dapat menambahkan barang baru ke toko berdasarkan antrian permintaan.

3.3.11 STORE REMOVE

STORE REMOVE merupakan fungsi yang digunakan untuk user dapat menghapus barang yang ada di toko.

3.3.12 HELP

HELP merupakan fungsi yang digunakan untuk user dapat melihat informasi mengenai command yang akan dipilihnya.

3.3.13 SAVE

SAVE merupakan fungsi yang digunakan untuk user dapat menyimpan state aplikasi terbaru ke dalam suatu file.

3.3.14 QUIT

QUIT merupakan fungsi yang digunakan untuk user dapat keluar dari sesi aplikasi PURRMART.

4 Algoritma-Algoritma Menarik

Di dalam program yang kami buat, terdapat 2 buah algoritma yang bisa dikategorikan sebagai unik. Algoritma yang unik ini adalah prosedur STORE REMOVE dan STORE LIST. Prosedur ini kami kategorikan sebagai algoritma menarik karena kami tidak menggabungkan kedua fitur ini di “main.c” tapi kami pisah di “console.c”

4.1 Prosedur STORE LIST

```
void STORELIST(array StoreList){
    if (StoreList.Neff == 0) {
        printf("TOKO KOSONG\n");
        return;
    }

    printf("List barang yang ada di toko : \n");
    for (int i = 0; i < StoreList.Neff; i++){
        printf("- ");
        PrintKata(StoreList.A[i]);
        printf("\n");
    }
}
```

Algoritma prosedur STORELIST berguna untuk menampilkan list barang yang terdapat di PURRMART. Algoritma ini menjadi sebuah algoritma menarik karena berada dalam file “console.c” dan tidak digabung di “main.c”

4.2 Prosedur STORE REMOVE

```
void STOREREMOVE(array *StoreList, ArrayDin *list){
    STORELIST(*StoreList);
    printf("\nNama barang yang akan dihapus : ");
    STARTSENTENCE();

    if (IsMemberArr(*StoreList, currentWord)) {
        for (int i = 0; i < StoreList->Neff; i++) {
            if (WordCompare(StoreList->A[i], currentWord)) {
                DeleteAt(StoreList, i);
                PrintKata(currentWord);
                printf(" telah berhasil dihapus.\n");
            }
            char barang[50];
            WordToString(currentWord, barang);
            if (SearchArrayDin(*list, barang)){
                DeleteAtL(list, i);
                list->Neff--;
                return;
            }
        }
    }
    else {
        printf("Toko tidak menjual ");
        PrintKata(currentWord);
        printf("\n");
    }
}
```

Algoritma prosedur STOREREMOVE berguna agar *user* dapat menghapuskan barang yang ada di STORE LIST. Algoritma ini merupakan algoritma yang menarik karena berada dalam file “console.c” yang bukan digabung pada “main.c”

5 Data Test

5.1 Data Test MAIN MENU

Pada tes ini dilakukan dilakukan pengujian untuk memastikan bahwa program dapat berjalan dengan baik dengan menampilkan tampilan awal dari aplikasi PURRMART. Tampilan awal menampilkan main menu yang berisikan pilihan START, LOAD, dan juga HELP. Cara melakukan kompilasi program ini dengan mengetikan. Adapun Gambar 6.1.1 yaitu pada saat tampilan menu pertama saat program dijalankan, gambar 6.1.2 yaitu pada saat tampilan menu saat start dijalankan, program akan masuk ke login menu, gambar 6.1.3 yaitu pada saat tampilan main menu saat *user* sudah berhasil login.

```
===== Selamat datang di =====
===== PURRMART =====
Perintah yang tersedia:
1. START
2. LOAD
3. HELP
4. QUIT
=====
```

Gambar 5.1.1 Tampilan Menu pertama saat program dijalankan

```
===== LOGIN MENU =====
Perintah yang tersedia:
1. LOGIN
2. REGISTER
3. HELP
4. QUIT
=====
```

Gambar 5.1.2 Tampilan Menu saat Start dijalankan, program akan masuk ke login menu

```
===== Main Menu =====
1. WORK
2. WORK CHALLENGE
3. STORE LIST
4. STORE REQUEST
5. STORE SUPPLY
6. STORE REMOVE
7. LOGOUT
8. SAVE
9. QUIT
10. HELP -> Untuk melihat masing-masing kegunaan command
=====
```

Gambar 5.1.3 Tampilan Main Menu saat user sudah berhasil login

5.2 Data Test START

Pada tes ini akan menjalankan program start dan memastikan bahwa fungsi berjalan dengan seharusnya. Adapun Gambar 5.2.1 merupakan tampilan pada saat tampilan pada saat file konfigurasi aplikasi berhasil dibaca.

```
===== Selamat datang di =====
===== PURRMART =====
|          Perintah yang tersedia:          |
|          1. START                         |
|          2. LOAD                         |
|          3. HELP                         |
|          4. QUIT                         |
|=====|
|          Masukkan pilihanmu: START      |
File konfigurasi aplikasi berhasil dibaca. PURRMART berhasil dijalankan.
```

Gambar 5.2.1 Tampilan saat file konfigurasi aplikasi berhasil dibaca

5.3 Data Test LOAD

Pada tes ini akan menjalankan input load dan memastikan bahwa fungsi berjalan dengan seharusnya. Adapun Gambar 5.3.1 merupakan tampilan pada saat save file tidak ditemukan dan PURRMART gagal dijalankan, Gambar 5.3.2 merupakan tampilan pada saat save file berhasil dibaca dan PURRMART berhasil dijalankan.

5.4 Data Test LOGIN

Pada tes ini akan menjalankan input login yaitu memasukkan username dan password dan apabila berhasil maka *user* akan bisa masuk ke aplikasi PURRMART. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 5.4.1 merupakan tampilan pada saat *user* berhasil login ke program, gambar 5.4.2 merupakan tampilan pada saat *user* salah memasukkan password, gambar 5.4.3 merupakan tampilan pada saat *user* salah memasukkan username, gambar 5.4.4 merupakan tampilan pada saat login gagal karena pengguna belum LOGOUT.

```
===== LOGIN MENU =====
|          Perintah yang tersedia:          |
|          1. LOGIN                         |
|          2. REGISTER                     |
|          3. HELP                         |
|          4. QUIT                         |
|=====|
|          Masukkan pilihanmu: LOGIN      |
Masukkan username: johndoe
Masukkan password: janedoe
Anda telah login ke PURRMART sebagai johndoe.
```

Gambar 5.4.1 Tampilan saat *user* berhasil login ke program

```
===== LOGIN MENU =====
|           Perintah yang tersedia:           |
|           1. LOGIN                          |
|           2. REGISTER                      |
|           3. HELP                          |
|           4. QUIT                          |
|=====|
|           Masukkan pilihanmu: LOGIN        |
|
Masukkan username: johndoe
Masukkan password: johndoe
Username atau password salah.
```

Gambar 5.4.2 Tampilan saat *user* salah memasukkan password

```
===== LOGIN MENU =====
|           Perintah yang tersedia:           |
|           1. LOGIN                          |
|           2. REGISTER                      |
|           3. HELP                          |
|           4. QUIT                          |
|=====|
|           Masukkan pilihanmu: LOGIN        |
|
Masukkan username: janedoe
Masukkan password: johndoe
Username atau password salah.
```

Gambar 5.4.3 Tampilan saat *user* salah memasukkan username

Gambar 5.4.4 Tampilan saat *user* gagal login karena belum LOGOUT

5.5 Data Test LOGOUT

Pada tes ini akan menjalankan logout yang dapat digunakan pengguna untuk menyelesaikan sebuah sesi pada aplikasi PURRMART. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 5.5.1 merupakan tampilan saat user berhasil logout dari program.

```
===== Main Menu =====
|                               |
| 1. WORK                      |
| 2. WORK CHALLENGE           |
| 3. STORE LIST                |
| 4. STORE REQUEST             |
| 5. STORE SUPPLY              |
| 6. STORE REMOVE              |
| 7. LOGOUT                    |
| 8. SAVE                      |
| 9. QUIT                      |
| 10. HELP -> Untuk melihat masing-masing kegunaan command |
|                               |
=====
Masukkan pilihanmu: LOGOUT
johndoe telah logout dari sistem PURRMART. Silakan REGISTER/LOGIN kembali untuk melanjutkan.
```

5.5.1 Tampilan saat *user* berhasil logout dari program

5.6 Data Test REGISTER

Pada tes ini akan menjalankan fitur register dimana *user* dapat mendaftarkan akun ke dalam sistem PURRMART. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 5.6.1 merupakan tampilan pada saat user berhasil mendaftarkan (*register*) akun, gambar 5.6.2 merupakan tampilan pada saat user gagal mendaftarkan (*register*) akun.

```
===== LOGIN MENU =====
| Perintah yang tersedia:     |
| 1. LOGIN                    |
| 2. REGISTER                  |
| 3. HELP                     |
| 4. QUIT                      |
|                               |
=====
Masukkan pilihanmu: REGISTER

Masukkan username: johndoe
Masukkan password: janedoe
Akun dengan username johndoe telah berhasil dibuat. Silakan LOGIN untuk melanjutkan.
```

5.6.1 Tampilan saat user berhasil mendaftarkan (*register*) akun

```
===== LOGIN MENU =====
| Perintah yang tersedia:     |
| 1. LOGIN                    |
| 2. REGISTER                  |
| 3. HELP                     |
| 4. QUIT                      |
|                               |
=====
Masukkan pilihanmu: REGISTER

Masukkan username: johndoe
Masukkan password: janedoe
Akun dengan username johndoe gagal dibuat. Silakan lakukan REGISTER ulang.
```

5.6.2 Tampilan saat user gagal mendaftarkan (*register*) akun

5.7 Data Test WORK

Pada tes ini akan menjalankan fitur WORK dimana *user* dapat memilih sejumlah pekerjaan dimana pada pekerjaan tersebut memiliki durasi dan pendapatan yang berbeda-beda. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 5.7.1 merupakan tampilan saat program berhasil bekerja.

```
===== Main Menu =====
| 1. WORK |
| 2. WORK CHALLENGE |
| 3. STORE LIST |
| 4. STORE REQUEST |
| 5. STORE SUPPLY |
| 6. STORE REMOVE |
| 7. LOGOUT |
| 8. SAVE |
| 9. QUIT |
| 10. HELP -> Untuk melihat masing-masing kegunaan command |
=====
Masukkan pilihanmu: 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: Mewing Specialist
Anda sedang bekerja sebagai Mewing Specialist... harap tunggu.
Pekerjaan selesai, +10000 rupiah telah ditambahkan ke akun Anda.
Uang anda sekarang adalah : 10100 rupiah
```

5.7.1 Tampilan program work saat berhasil bekerja

5.8 Data Test WORK CHALLENGE

Pada tes ini akan menjalankan fitur WORK CHALLENGE dimana *user* dapat memilih sejumlah challenge yang dapat digunakan untuk mendapatkan uang. Challenge yang diberikana adalah Tebak Angka dan juga WORDLE. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 6.8.1 merupakan tampilan pada saat *challenge* tebak angka berhasil ditamatkan, gambar 6.8.2 merupakan tampilan pada saat *user* berhasil menamatkan *challenge* WORDL399, gambar 6.8.3 merupakan tampilan pada saat *user* gagal menamatkan challenge WORDL399.

```

Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL399 (biaya main=500)
Masukkan challenge yang hendak dimainkan: 1

Uang Anda tersisa 9900.

Tebak Angka (1-100) dengan kesempatan tersisa 10.
Tebak Angka = 10
Tebakanmu lebih besar!
Sisa kesempatan adalah 9.

Tebak Angka = 5
Tebakanmu benar! 450 rupiah telah ditambahkan ke akun Anda
Uang anda sekarang adalah = 10350.

```

Gambar 5.8.1 Tampilan saat *user* berhasil menamatkan *challenge* Tebak Angka

```

Sisa kesempatan adalah 9.

Tebak Angka = 20
Tebakanmu lebih kecil!
Sisa kesempatan adalah 8.

Tebak Angka = 30
Tebakanmu lebih kecil!
Sisa kesempatan adalah 7.

Tebak Angka = 40
Tebakanmu lebih kecil!
Sisa kesempatan adalah 6.

Tebak Angka = 50
Tebakanmu lebih kecil!
Sisa kesempatan adalah 5.

Tebak Angka = 60
Tebakanmu lebih kecil!
Sisa kesempatan adalah 4.

Tebak Angka = 70
Tebakanmu lebih kecil!
Sisa kesempatan adalah 3.

Tebak Angka = 80
Tebakanmu lebih kecil!
Sisa kesempatan adalah 2.

Tebak Angka = 90
Tebakanmu lebih besar!
Sisa kesempatan adalah 1.

Tebak Angka = 100
Tebakanmu lebih besar!
Sisa kesempatan adalah 0.

Kesempatan Anda sudah habis. Angka yang benar adalah 83. Uang Anda tersisa 9950

```

Gambar 5.8.2 Tampilan saat *user* gagal menamatkan *challenge* Tebak Angka


```
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL399 (biaya main=500)
Masukkan challenge yang hendak dimainkan: 2

Uang Anda tersisa 9700.

SELAMAT DATANG DI WORDL3, ANDA MEMILIKI 6 PELUANG UNTUK MENEBAK KATA!

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

Masukkan kata tebakan Anda: ADIEU
Hasil:
A%D%I%E%U*
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -

A%D%I%E%U*
- - - - -
- - - - -
- - - - -

Masukkan kata tebakan Anda: TRULY
Hasil:
A%D%I%E%U*
T%R%U*L*Y
- - - - -
- - - - -

Masukkan kata tebakan Anda: TRULY
Hasil:
A%D%I%E%U*
T%R%U*L*Y
- - - - -
- - - - -
- - - - -

A%D%I%E%U*
T%R%U*L*Y
- - - - -
- - - - -
- - - - -

Masukkan kata tebakan Anda: LUCKY
Hasil:
A%D%I%E%U*
T%R%U*L*Y
L U C K Y
- - - - -
- - - - -
- - - - -

Selamat, Anda menang!
+1500 rupiah telah ditambahkan ke akun Anda.
Uang Anda sekarang: 11200
```

Gambar 5.8.3 Tampilan saat user berhasil menang game wordle (WORDL399)

```
Masukkan kata tebakan Anda: PLUCK
Hasil:
A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
- - - - -
- - - - -
- - - - -

A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
- - - - -
- - - - -

Masukkan kata tebakan Anda: SERIN
Hasil:
A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
S*E%R%I*N%
- - - - -
- - - - -
- - - - -

A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
S*E%R%I*N%
- - - - -
- - - - -

Masukkan kata tebakan Anda: TRULY
Hasil:
A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
S*E%R%I*N%
T%R%U%L%Y%
- - - - -

A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
S*E%R%I*N%
T%R%U%L%Y%

Masukkan kata tebakan Anda: SLICK
Hasil:
A%D%I*E%U%
O*M%B%R%E%
P%L%U%C K%
S*E%R%I*N%
T%R%U%L%Y%
S*L*I*C K%

Maaf, Anda kalah! Kata yang benar adalah: CISCO
Uang Anda tersisa: 10700
```

Gambar 5.8.4 Tampilan saat user kalah bermain W0RDL399

5.9 Data Test STORE LIST

Pada tes ini akan menjalankan fitur STORE LIST dimana *user* dapat melihat barang-barang unik apa saja yang ada di toko. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya.

```
List barang yang ada di toko :  
- Buku Baru  
- Pensil  
- Tas  
- Jam Tangan Mahal  
- Gantungan Kunci
```

Gambar 5.9.1 Tampilan list barang-barang

5.10 Data Test STORE REQUEST

Pada tes ini akan menjalankan fitur STORE REQUEST dimana *user* dapat meminta penambahan barang baru ke toko. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 5.10.1 adalah saat pengguna akan meminta barang, gambar 5.10.2 adalah saat barang yang diminta ternyata sudah ada di toko, gambar 5.10.3 adalah pada saat barang yang sama sudah ada di antrian.

```
Masukkan pilihanmu: STORE REQUEST  
  
Nama barang yang diminta: AWM  
  
===== Main Menu =====  
|                               |  
| 1. WORK                     |  
| 2. WORK CHALLENGE          |  
| 3. STORE LIST               |  
| 4. STORE REQUEST            |  
| 5. STORE SUPPLY             |  
| 6. STORE REMOVE             |  
| 7. LOGOUT                   |  
| 8. SAVE                     |  
| 9. QUIT                     |  
| 10. HELP -> Untuk melihat masing-masing kegunaan command |  
|                               |  
===== Masukkan pilihanmu: STORE REQUEST
```

Gambar 5.10.1 Tampilan saat pengguna akan meminta barang

```
Masukkan pilihanmu: STORE REQUEST  
Nama barang yang diminta: MP  
Barang dengan nama yang sama sudah tersedia di toko!
```

Gambar 5.10.2 Tampilan saat barang yang diminta sudah ada di toko

```
Masukkan pilihanmu: STORE REQUEST  
Nama barang yang diminta: AWM  
Barang dengan nama yang sama sudah tersedia di antrian!
```

Gambar 5.10.3. Tampilan saat barang yang sama sudah ada di antrian

5.11 Data Test STORE SUPPLY

Pada tes ini akan menjalankan fitur STORE SUPPLY dimana *user* dapat menambahkan barang baru ke toko berdasarkan antrian permintaan. Pengguna dapat menerima, menunda, atau menolak permintaan. Adapun gambar 5.11.1 merupakan tampilan saat barang diterima, gambar 5.11.2.. merupakan tampilan saat barang ditunda, dan gambar 5.11.3., merupakan tampilan saat barang ditolak. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya.

```
Masukkan pilihanmu: STORE SUPPLY  
Apakah kamu ingin menambahkan barang M416: Terima  
Harga barang: 100
```

Gambar 5.11.1 Tampilan saat barang diterima

```
M416 dengan harga 100 telah ditambahkan ke toko.  
Apakah kamu ingin menambahkan barang M16A2: Tunda
```

Gambar 5.11.2 Tampilan saat barang ditunda

```

M16A2 dikembalikan ke antrian.
Apakah kamu ingin menambahkan barang M4: Tolak

M4 dihapuskan dari antrian.
Apakah kamu ingin menambahkan barang M16A2: Tolak

```

Gambar 5.11.3 Tampilan saat barang ditolak

5.12 Data Test STORE REMOVE

Pada tes ini akan menjalankan fitur STORE REMOVE dimana *user* dapat menghapus barang yang ada di toko. Fitur STORE REMOVE akan menerima input barang mana yang akan dihapus. Adapun gambar 5.12.1 merupakan tampilan saat barang yang dihapus berhasil dihapus, dan gambar 5.12.2 merupakan tampilan saat barang yang dihapus tidak berhasil dihapus karena tidak ada barang tersebut di toko. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya.

```

List barang yang ada di toko :
- Buku Baru
- Pensil
- Tas
- Jam Tangan Mahal
- Gantungan Kunci

Nama barang yang akan dihapus : Tas
Tas telah berhasil dihapus.
List barang yang ada di toko :
- Buku Baru
- Pensil
- Jam Tangan Mahal
- Gantungan Kunci

```

Gambar 5.12.1 Tampilan barang yang berhasil dihapus

```

6. STORE REMOVE
7. LOGOUT
8. SAVE
9. QUIT
10. HELP -> Untuk melihat masing-masing kegunaan command
=====
Masukkan pilihanmu: STORE REMOVE

List barang yang ada di toko :
- MP
- AKM

Nama barang yang akan dihapus : AK
Toko tidak menjual AK

```

Gambar 5.12.2 Tampilan barang yang gagal dihapus karena tidak ada barang tersebut di toko

5.13 Data Test HELP

Pada tes ini akan menjalankan fitur HELP dimana *user* dapat menampilkan daftar command yang akan dieksekusi pada masing-masing page. Terdapat HELP pada *welcome menu*, HELP pada *login menu*, dan HELP pada *main menu*. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya Adapun gambar ... merupakan tampilan saat fitur HELP dijalankan pada *welcome menu* (tampilan pertama kali), gambar ... merupakan tampilan saat fitur HELP dijalankan pada *login menu* (sebelum login), dan gambar ... merupakan tampilan saat fitur HELP dijalankan pada *main menu* (setelah login berhasil).

```
===== Selamat datang di =====
===== PURRMART =====
|          Perintah yang tersedia:          |
|          1. START                         |
|          2. LOAD                         |
|          3. HELP                         |
|          4. QUIT                         |
|-----|
|          Masukkan pilihanmu: HELP        |
|-----|
|===== HELP MENU =====|
|          1. START -> Untuk masuk sesi baru |
|          2. LOAD -> Untuk memulai sesi berdasarkan file konfigurasi |
|          3. QUIT -> Untuk keluar dari program |
|-----|
```

Gambar 6.13.1 Tampilan Help Menu pada tampilan awal masuk program

```
===== LOGIN MENU =====
|          Perintah yang tersedia:          |
|          1. LOGIN                         |
|          2. REGISTER                     |
|          3. HELP                         |
|          4. QUIT                         |
|-----|
|          Masukkan pilihanmu: HELP        |
|-----|
|===== HELP MENU =====|
|          1. REGISTER -> Untuk melakukan pendaftaran akun baru |
|          2. LOGIN -> Untuk masuk ke dalam akun dan memulai sesi |
|          3. QUIT -> Untuk keluar dari program |
|-----|
```

Gambar 6.13.2 Tampilan Help Menu pada tampilan Login Menu

```

===== Main Menu =====
1. WORK
2. WORK CHALLENGE
3. STORE LIST
4. STORE REQUEST
5. STORE SUPPLY
6. STORE REMOVE
7. LOGOUT
8. SAVE
9. QUIT
10. HELP -> Untuk melihat masing-masing kegunaan command
=====
Masukkan pilihanmu: HELP

===== HELP MENU =====
1. WORK -> Untuk bekerja
2. WORK CHALLENGE -> Untuk mengerjakan challenge
3. STORE LIST -> Untuk melihat barang-barang di toko
3. QUIT -> Untuk keluar dari program
4. STORE REQUEST -> Untuk meminta penambahan barang
5. STORE SUPPLY -> Untuk menambahkan barang dari permintaan
6. STORE REMOVE -> Untuk menghapus barang
7. LOGOUT -> Untuk keluar dari sesi
8. SAVE -> Untuk menyimpan state ke dalam file
9. QUIT -> Untuk keluar dari program
=====

```

Gambar 6.13.3 Tampilan Help Menu pada Main Menu (sesudah berhasil login)

5.14 Data Test SAVE

Pada tes ini akan menjalankan fitur SAVE dimana *user* dapat menyimpan state aplikasi terbaru ke dalam suatu file. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya. Adapun gambar 6.14.1 merupakan tampilan saat fitur SAVE dijalankan dan file yang akan disave berhasil disimpan.

```

3. STORE LIST
4. STORE REQUEST
5. STORE SUPPLY
6. STORE REMOVE
7. LOGOUT
8. SAVE
9. QUIT
10. HELP -> Untuk melihat masing-masing kegunaan command
=====
Masukkan pilihanmu: SAVE test.txt

src\save\test.txt
Save file berhasil disimpan.

```

Gambar 6.14.1 Tampilan saat SAVE dijalankan dan file berhasil disimpan

5.15 Data Test QUIT

Pada tes ini akan menjalankan fitur QUIT dimana *user* dapat keluar dari sesi aplikasi PURRMART. Setelah itu *user* akan diminta apakah ingin menyimpan data sesi tersebut atau tidak. Adapun gambar 6.15.1 merupakan tampilan saat fitur QUIT dijalankan dan user tidak ingin keluar dari aplikasi, gambar 6.15.2 merupakan tampilan saat fitur QUIT dijalankan dan user ingin keluar dari aplikasi. Tes ini memastikan bahwa seluruh fungsi berjalan dengan seharusnya.

```

Masukkan pilihanmu: QUIT

Apakah anda ingin menyimpan data sesi sekarang (Y/N)? N

Exiting program...

```

Gambar 6.15.1 Tampilan saat *user* ingin keluar dari program dan tidak ingin save progressnya

```

Perintah yang tersedia:
1. LOGIN
2. REGISTER
3. HELP
4. QUIT
=====
Masukkan pilihanmu: QUIT

Apakah anda ingin menyimpan data sesi sekarang (Y/N)? Y
Ketik 'SAVE' diikuti nama file: 

```

Gambar 6.15.2 Tampilan saat *user* ingin keluar dari program dan ingin save progressnya.

6 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1.	Fitur MAIN MENU	Memeriksa apakah program bisa menampilkan tampilan awal dari aplikasi PURRMART	Pada saat menjalankan program, akan keluar <i>main menu</i>	Data Test 6.1	Memunculkan main menu yang terdapat 3 pilihan yaitu START, LOAD GAME, dan HELP	Sesuai yang diharapkan
2.	Fitur START	Memeriksa apakah fitur START bisa berjalan dengan baik dan program PURRMART bisa dimulai dan dijalankan	Memasukkan <i>command START</i> pada main menu	Data Test 6.2	Memunculkan fitur pada PURRMART yang dapat dipilih kemudian akan diminta <i>input</i> fitur.	Sesuai yang diharapkan

3.	Fitur LOAD	Memeriksa apakah fitur LOAD bisa berjalan dengan baik	Memasukkan <i>command</i> LOAD untuk memulai file yang pernah dijalankan	Data Test 6.3	Menjalankan progres yang telah disimpan	Sesuai yang diharapkan
4.	Fitur LOGIN	Memeriksa apakah fitur LOGIN bisa berjalan dengan baik dan mengeluarkan output sesuai yang diharapkan	Memasukkan command LOGIN yaitu username dan password agar user dapat masuk ke akunnya	Data Test 6.4	User dapat berhasil login ke dalam program	Sesuai yang diharapkan
5.	Fitur LOGOUT	Memeriksa apakah fitur LOGOUT dapat berjalan dengan baik dan mengeluarkan final state yang seharusnya	Memasukkan command LOGOUT agar user dapat keluar dari akunnya	Data Test 6.5	User dapat berhasil logout dari program	Sesuai yang diharapkan
6.	Fitur REGISTER	Memeriksa apakah fitur REGISTER dapat berjalan dengan baik dan input username dan password dapat dimasukkan ke dalam toko.	Memasukkan command REGISTER agar user dapat mendaftarkan akunnya pada database program	Data Test 6.6	User dapat berhasil mendaftarkan akun ke program	Sesuai yang diharapkan
7.	Fitur WORK	Memeriksa apakah fitur WORK dapat berjalan dengan baik dan output yang dihasilkan sesuai yang diharapkan.	Memasukkan command WORK agar user dapat mendapatkan uang dengan bekerja dengan berbagai peran	Data Test 6.7	User dapat berhasil menjalankan program dan mendapatkan uang dengan menjalankan program work.	Sesuai yang diharapkan

8.	Fitur WORK CHALLENGE	Memeriksa apakah fitur WORK CHALLENGE dapat berjalan dengan baik dan kedua game dapat berjalan dengan lancar.	Memasukkan command WORK CHALLENGE agar user dapat menggunakan uang dari WORK untuk bermain CHALLENGE yang jika user menamatkan CHALLENGE akan mendapatkan hadiah berupa uang,	Data Test 6.8	User dapat berhasil menjalankan program work challenge dan berhasil memainkan work challenge	Sesuai yang diharapkan
9.	Fitur STORE LIST	Memeriksa apakah fitur STORE LIST dapat berjalan dengan baik dan dapat menampilkan barang-barang yang ada di toko.	Memasukkan <i>command</i> STORE LIST untuk menampilkan barang-barang yang ada pada toko	Data Test 6.9	Menampilkan barang-barang yang ada pada toko	Sesuai yang diharapkan
10.	Fitur STORE REQUEST	Memeriksa apakah fitur STORE REQUEST dapat berjalan dengan baik dan dapat menampilkan input yang meminta barang baru yang akan dimasukkan user.	Memasukkan command STORE REQUEST untuk user dapat meminta barang baru yang akan dimasukkan.	Data Test 6.10	User dapat berhasil meminta barang baru ke dalam program	Sesuai yang diharapkan
11.	Fitur STORE SUPPLY	Memeriksa apakah fitur STORE SUPPLY dapat berjalan dengan baik dan input user dapat diterima dengan baik oleh fitur.	Memasukkan comand STORE SUPPLY untuk user dapat menambahkan barang sesuai dengan antrian.	Data Test 6.11	User dapat berhasil mendapatkan barang sesuai dengan antrian.	Sesuai yang diharapkan

12.	Fitur STORE REMOVE	Memeriksa apakah fitur STORE REMOVE dapat berjalan dengan baik dan input user dapat diterima dengan baik oleh fitur.	Memasukkan <i>command</i> STORE REMOVE untuk menghapus barang	Data Test 6.12	Barang yang dimasukkan pada input berhasil terhapus dari STORE LIST	Sesuai yang diharapkan
13.	Fitur HELP	Memeriksa apakah fitur HELP dapat berjalan dengan baik dan output yang dihasilkan sesuai dengan yang diharapkan.	Memasukkan command HELP untuk user dapat melihat informasi command pada masing-masing menu	Data Test 6.13	User dapat berhasil memanggil fungsi HELP dan melihat kegunaan masing-masing command	Sesuai yang diharapkan
14.	Fitur SAVE	Memeriksa apakah fitur SAVE dapat berjalan dengan baik dan fitur dapat menyimpan file dengan yang seharusnya.	Memasukkan command SAVE untuk menyimpan state pada file.	Data Test 6.14	User dapat berhasil menyimpan state pada file	Sesuai yang diharapkan
15.	Fitur QUIT	Memeriksa apakah fitur QUIT dapat berjalan dengan baik dan <i>user</i> dapat keluar dari program PURRMART.	Memasukkan command QUIT untuk keluar dari program dan meminta user untuk melakukan save pada program	Data Test 6.15	User dapat berhasil keluar dari program setelah menyelesaikan program.	Sesuai yang diharapkan

7 Pembagian Kerja dalam Kelompok

Berikut merupakan penjelasan pembagian kerja dalam kelompok.

No	Fitur/ADT	Nim Coder	Nim Tester
1.	ADT Array	18223059	18221059, 18221004, 18223021, 18223019, 18223089
2.	ADT Mesin Kata	18223021,18223059	1823021, 18223059, 18221004, 18223019, 18223089
3.	ADT Mesin Karakter	18223021,18223059	18223021, 18223059, 18221004, 18223019, 18223089
4.	ADT Queue	18223059	18223059, 18221004, 18223021, 18223019, 18223089
5.	ADT List	18223059	18223059, 18221004, 18223021, 18223019, 18223089
6.	Main Menu	18223021	18223021, 18221004, 18223019, 18223059,18223089
7.	START	18223059	18223059, 18221004, 18223021, 18223019, 18223089
8.	LOAD	18221004	18221004, 18223021, 18223019, 18223059, 18223089
9.	LOGIN	18223089	18223089, 18221004, 18223021, 18223019 18223059
10/	LOGOUT	18223089	18223089, 18221004, 18223021, 18223019 18223059
11.	REGISTER	18223089	18223089, 18221004, 18223021, 18223019

			18223059
12.	WORK	18223021	18223021, 18221004, 18223019, 18223059, 18223089
13.	WORK CHALLENGE	18223021 (Tebak Angka), 18223019 (WORDL399)	18223021, 18221004, 18223019, 18223059, 18223089
14.	STORE LIST	18221004	18221004, 18223021, 18223019, 18223059, 18223089
15.	STORE REQUEST	18221004	18221004, 18223021, 18223019, 18223059, 18223089
16.	STORE SUPPLY	18223059	18223059, 18221004, 18223021, 18223019, 18223059, 18223089
17.	STORE REMOVE	18223059	18223059, 18221004, 18223021, 18223019, 18223059, 18223089
18.	HELP	18223021	18223059, 18221004, 18223021, 18223019, 18223059, 18223089
19.	SAVE	18223089	18223089, 18221004, 18223021, 18223019, 18223059
20.	QUIT	18223019	18223019, 18221004, 18223021, 18223059, 18223089

8 Lampiran

8.1 Deskripsi Tugas Besar 1

PURRMART sebuah sistem e-commerce yang dibuat untuk membantu OWCA membeli suplai perang dari Toko Borma secara digital. Sistem ini menggantikan transaksi tatap muka dengan fitur-fitur seperti pemesanan barang digital, manajemen stok, efisiensi operasi. PURRMART menjadi solusi logistik OWCA untuk tetap bertahan dan mencapai kemenangan tanpa hambatan.

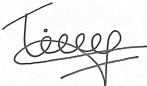
8.2 Notulen Rapat






**Form Asistensi Tugas Besar
IF2111/Algoritma dan Struktur Data STI
Sem. 1 2024/2025**

No. Kelompok/Kelas : 05/01
Nama Kelompok :
Anggota Kelompok (Nama/NIM) :
1. Timothy Marvine/18223021
2. Muhammad Naufal Fathan/18223059
3. Adam Joaquin Girsang/18223089
4. A.Nurul Aqeela Amin/18223019
5. Khairunnisa Hurun 'Iin/18221004

Asisten Pembimbing : Jonathan Arthurito Aldi Sinaga

Asistensi I

Tanggal : 23 November 2024	Catatan Asistensi: 1. Progres : Sudah berprogres, tapi belum mengimplementasikan mesin kata 2. Pertanyaan : START , untuk load file config, apakah boleh menggunakan fgets dan scanf? - Tidak diperbolehkan REQUEST dan SUPPLY , apakah bisa dihandle oleh user yang sama atau hanya bisa untuk role user tertentu? - Setiap user bisa melakukan setiap command
Tempat : Google Meet	
Kehadiran Anggota Kelompok: 1 18223021 Timothy Marvine  2 18223059 Muhammad Naufal Fathan	

 3 18223090 Adam Joaquin Girsang  4 18223019 A.Nurul Aqeela Amin  5 18221004 Khairunnisa Hurun 'Iin 	<p>Dokumen, apakah perbedaan bagian pembagian tugas dan <i>log activity</i>?</p> <ul style="list-style-type: none"> - Log activity untuk aktivitas lebih detail setiap ada progres <p>Bagaimana cara menggabungkan program?</p> <ul style="list-style-type: none"> - Include header yang digunakan di file main.c <p>Apakah kegunaan driver?</p> <ul style="list-style-type: none"> - Untuk membuktikan ADT yang dibentuk dapat dijalankan <p>3. Summary : Coba rapihin foldernya, saran dikelarkan besok supaya tidak panik, selesaikan dokumennya</p>
	<p>Tanda Tangan Asisten:</p> 

8.3 Log Activity Anggota Kelompok

Berikut merupakan penjelasan pembagian kerja dalam kelompok.

No.	Tanggal	Kegiatan	Hasil
1.	13-11-2024	<ul style="list-style-type: none"> - Pembagian tugas - Pembuatan Github 	Masing-masing anggota menerima pembagian

			tugas dan mulai mengerjakan bagian masing-masing pada Github
2.	21-11-2024	- <i>Offline meeting</i>	Membagikan progres masing-masing dan mengerjakan bersama
3.	22-11-2024	- <i>Offline meeting</i>	Membagikan progres masing-masing dan mengerjakan bersama
4.	23-11-2024	- <i>Offline meeting</i> - Asistensi I	Membagikan progres masing-masing dan mengerjakan bersama serta melakukan penyesuaian berdasarkan hasil asistensi
5.	24-11-2024	- <i>Offline meeting</i>	Membagikan progres dan melakukan integrasi untuk finalisasi serta melengkapi dokumen
6.	25-11-2024	- <i>Offline meeting</i>	Melanjutkan tahap integrasi untuk finalisasi serta melengkapi dokumen dan pengumpulan