**FOOD ORDER MANAGEMENT SYSTEM**

By

PULKIT CHAUHAN 17BEC1026

SAQLAIN RAHMAN 17BEC1066

ARYAN SHARMA 17BEC1152

A project report submitted to

**Dr. R. Rajalakshmi**

**SCHOOL OF ELECTRONICS ENGINEERING**

in partial fulfilment of the requirements for the course of

**CSE 2003 – DATA STRUCTURE AND ALGORITHM**

in

**B.Tech. (Electronics and Communication Engineering)**



**VELLORE INSTITUTE OF TECHNOLOGY, CHENNAI**

**Vandalur – Kelambakkam Road, Chennai – 600127**

**MARCH – 2019**

**BONAFIDE CERTIFICATE**

Certified that this project report entitled “FOOD ORDER MANAGEMENT SYSTEM**”** is a bonafide work of **PULKIT CHAUHAN (17BEC1026),**

**SAQLAIN RAHMAN(17BEC1066), ARYAN SHARMA(17BEC1152)**

who carried out the “J”-Project work under my supervision and guidance for CSE 2003 – Data structure and algorithm

**Dr. R. Rajalakshmi**

School of Electronics Engineering (SENSE),

VIT University, Chennai

Chennai – 600 127.

**ABSTRACT**

Food order management system is system where customers can buy food with cash or credit card and this system have an admin panel where admin can add food, delete food and count food which will be shown on the screen. Using data structures like linked lists and Trie, we can make the system of food order management reliable.

Also, system shows the total number of order in that day with transaction history to admin. And there is extra feature of Backup System where admin can view the history of payment of any day through total money and Card list with payment details.

**ACKNOWLEDGEMENT**

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. R. Rajalakshmi**, School of Electronics Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Sreedevi.V.T,** Dean of the Schools of Electrical Engineering (SELECT) and Electronics Engineering (SENSE), VIT University Chennai, for extending the facilities of the School towards our project and for her unstinting support.

We express our thanks to our Programme Chair **Dr. Vetrivelan (for B.Tech-ECE)** for her support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

**PULKIT CHAUHAN**

**SAQLAIN RAHMAN**

**ARYAN SHARMA**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TOPIC** | **PAGE** |
| **1** | Introduction | 6 |
| **2** | Approach | **7** |
| **3**  **4**  **5**  **6**  **7** | Algorithm  Software required  Simulation Code  Output  Additional Work | 8-9  10  11-38  39-42  43 |

8 Conclusion 44

9 References 45

**INTRODUCTION**

**OBJECTIVES AND GOALS**

Food order management system is system where customers can buy food with cash or credit card and this system have an admin panel where admin can add food, delete food and count food which will be shown on the screen. Also, system shows the total number of order in that day with transaction history to admin. And there is extra feature of Backup System where admin can view the history of payment of any day through total money and Card list with payment details.

Using Trie data structures , customer can search his/her favorite food items.

If food item is present and in stock, he/she can directly order the food.

**APPROACH**

HOW TRIE FUNCTION WORKS:

Every node of Trie consists of multiple branches. Each branch represents a possible character of keys.

Every character of input key is inserted as an individual Trie node. Note that the *children* is an array of pointers (or references) to next level trie nodes.

The key character acts as an index into the array *children*. If the input key is new or an extension of existing key, we need to construct non-existing nodes of the key, and mark end of word for last node. If the input key is prefix of existing key in Trie, we simply mark the last node of key as end of word. The key length determines Trie depth.

Searching for a key is similar to insert operation, however we only compare the characters and move down. The search can terminate due to end of string or lack of key in trie. In the former case, if the *isEndofWord* field of last node is true, then the key exists in trie. In the second case, the search terminates without examining all the characters of key, since the key is not present in trie.

**ALGORITHM**

**Data Structure: Trie**

* Using Trie, search complexities can be brought to optimal limit (key length).
* struct TrieNode
* {
* struct TrieNode \*children[ALPHABET\_SIZE];
* bool isEndOfWord;
* };
* struct TrieNode \*getNode(void)
* {
* struct TrieNode \*pNode = NULL;
* pNode = (struct TrieNode \*)malloc(sizeof(struct TrieNode));
* if (pNode)
* {
* int i;
* pNode->isEndOfWord = false;
* for (i = 0; i < ALPHABET\_SIZE; i++)
* pNode->children[i] = NULL;
* }
* return pNode;
* }

**Data Structure: linked list**

* Int main(){
* Void cls(){
* Void echo(char print[]){
* Void br(int line){
* Void pre(int tab){
* Void span(int space){
* Void main\_menu(){
* Void insertend(int data, char foodname[25], int quantity, float price){
* Void insertfirst(int data, char foodname[25], int quantity, float price){
* Void insertmid(int pos, int data, char foodname[25], int quantity, float price){
* Void deletefood(int serial){
* Void updatefood(int udata, int uquantity){

**SOFTWARE REQUIRED**

**C Language:**

Features:

1. Linked list to add and delete item from the list .
2. Can view card details of payment in admin panel.
3. Backup Option for the data stored.
4. Can pay through card and cash both.
5. Customer can search his favorite item using Trie

**SOFTWARE PROGRAM**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<string.h>

#include<windows.h>

#include<time.h>

#include<string.h>

#include <stdbool.h>

#define ARRAY\_SIZE(a) sizeof(a)/sizeof(a[0])

#define ALPHABET\_SIZE (26)

#define CHAR\_TO\_INDEX(c) ((int)c - (int)'a')

void insertfirst(int data, char foodname[25], int quantity, float price);

void insertmid(int pos, int data, char foodname[25], int quantity, float price);

void insertend(int data, char foodname[25], int quantity, float price);

void updatefood(int udata, int uquantity);

bool searchfood(const char \*foodnme,struct TrieNode \*root);

void insert(struct TrieNode \*root,char key[25]);

bool search(struct TrieNode \*root,char key[25]);

void foodlist();

void order\_view(int order, int quantity, int or\_no);

void main\_menu();

void deletefood(int serial);

int countitem();

void cls();

void echo(char print[]);

void br(int line);

void pre(int tab);

void span(int space);

//void ccolor(int clr);

void pwellcome();

void loadingbar();

void middle1(void);

void middtab1(void);

void backuploader(void);

struct Node{

char foodname[25];

int quantity;

float price;

int data;

struct Node \*next;

};

typedef struct Node node ;

node \*head, \*list;

struct TrieNode

{

struct TrieNode \*children[ALPHABET\_SIZE];

bool isEndOfWord;

};

struct TrieNode \*getNode(void)

{

struct TrieNode \*pNode = NULL;

pNode = (struct TrieNode \*)malloc(sizeof(struct TrieNode));

if (pNode)

{

int i;

pNode->isEndOfWord = false;

for (i = 0; i < ALPHABET\_SIZE; i++)

pNode->children[i] = NULL;

}

return pNode;

}

int main(){

struct TrieNode \*root = getNode();

system("title ..................FOOD ORDER MANAGEMENT SYSTEM.......................");

system("mode con: cols=80 lines=30");

//loadingbar();

cls();

pwellcome();

Sleep(300);

cls();

int c=0; int any;

int cardno[100];

float cardmoney[100];

float totalmoney = 0;

int total\_order[100];

int order\_quantity[100];

int order=0;

int uquantity;

int citem;

char food[25];

int i,len=9;

head = NULL;

insertfirst(5,"Burger",23,120.23);

insertend(6,"Pizza",13,100.67);

insertend(1,"Hot Cake",8,720.83);

insertend(2,"Coffie",46,70.23);

insertend(3,"Ice-Cream",46,70.23);

insertend(4,"Sandwich",34,60.23);

insertend(7,"Grill",7,520.29);

insertend(8,"Nun-Bread",121,35.13);

insertend(9,"Cold Drinks",73,20.13);

mainmenu:

br(1);

main\_menu();

int main\_menu\_choice;

br(1); pre(4); fflush(stdin); scanf("%d",&main\_menu\_choice);

if((main\_menu\_choice >=1 && main\_menu\_choice <=4)){

if(main\_menu\_choice == 1){

foodlist:

cls();

printf("=> 0. Main Menu ");

foodlist();

}

else if( main\_menu\_choice == 2){

adminpanelchoice:

int admin\_panel\_choice;

cls(); middle1() ; pre(4); printf("1. Main Menu\n\n\t"); Sleep(300);

printf("Please Enter Password or ( 1 to Back in Main Menu ) : ");

fflush(stdin); scanf("%d",&admin\_panel\_choice);

if(admin\_panel\_choice==123321){

node \*temp;

temp = list;

adminchoise:

cls(); br(5); pre(4); echo("You are on Admin Pannel\n\n");

pre(4);

printf(" 1. Total Cash Today \n\n");Sleep(250);pre(4);

printf(" 2. View Card Pay \n\n");Sleep(250);pre(4);

printf(" 3. Add Food \n\n");Sleep(250);pre(4);

printf(" 4. Delete Food \n\n");Sleep(250);pre(4);

printf(" 5. Instant Food List \n\n");Sleep(250);pre(4);

printf(" 6. Item Counter \n\n");Sleep(250);pre(4);

printf(" 7. Backup System\n\n");Sleep(250);pre(4);

printf(" 8. Instant Order Preview\n\n");Sleep(250);pre(4);

printf(" 0. Main Menu ");Sleep(250);

int adminchoise;

fflush(stdin); scanf("%d",&adminchoise);

if(adminchoise==1){

cls(); middle1(); pre(4); printf("Todays Total Cash : %0.2f \n",totalmoney);

Sleep(2000);

goto adminchoise;

}

else if(adminchoise==2){

if(c!=0){

cls(); br(3); pre(4);

printf(" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");pre(4);

printf("| Card NO. | Money $ |\n");pre(4);

printf("------------------------------\n");pre(4);

for(int z=1; z<=c;z++){

printf("| %d | %0.2f |\n",cardno[z],cardmoney[z]);pre(4);

printf("------------------------------\n");pre(4);

Sleep(150);

}

Sleep(1500);

}

if(c==0){

cls(); middle1(); pre(4);

printf("No Card History\n");}

Sleep(1500);

goto adminchoise;

}

else if(adminchoise==3){

foodadd:

cls();

char ffoodname[25];

int fquantity;

int fdata;

float fprice;

int fposi;

br(3);pre(4); printf(" Enter Food Name : ");

fflush(stdin); scanf("%[^\n]s",ffoodname);

fquantity:

fflush(stdin);

br(2);pre(4); printf(" Enter Food Quantity : ");

scanf("%d",&fquantity); fflush(stdin);

foodserial:

br(2);pre(4); printf(" Enter Food Serial : ");

scanf("%d",&fdata);

node \*exist;

exist = list;

while(exist->data!=fdata){

if(exist->next==NULL){

break;

}

exist=exist->next;

}

if(exist->data==fdata){

cls(); br(5);pre(3); printf(" Food Serial Already Exist, Please Re-Enter "); Sleep(2000);

goto foodserial;

}

fprice:

fflush(stdin);

br(2);pre(4); printf(" Enter Food Price : ");fflush(stdin);

scanf("%f",&fprice);

br(2);pre(4); printf("Submiting your data");for(int cs=0;cs<4;cs++){printf(" .");Sleep(500);}

insertend(fdata,ffoodname,fquantity,fprice);

br(2);pre(4); printf("Adding Food Successfull\n");

Sleep(2000);

goto adminchoise;

}

else if(adminchoise==4){

cls();

middle1();pre(2);

printf("Enter Serial No of the Food To Delete : ");

fdelete:

int fdelete;

fflush(stdin); scanf("%d",&fdelete);

node \*temp;

temp=list;

while(temp->data != fdelete){

temp = temp->next;

}

if(temp->data==fdelete){

deletefood(fdelete);

}

else{

br(2); pre(2); printf("Please Enter Correct Number : "); Sleep(500);

goto fdelete;

}

goto adminchoise;

}

else if(adminchoise==5){

cls(); foodlist(); Sleep(1000);

br(2);pre(4); printf("1. <-- back \n\n");pre(5);

fflush(stdin); scanf("%d",&any);

goto adminchoise;

}

else if(adminchoise==6){

citem = countitem();

cls();

for(int cs=1;cs<=citem;cs++){

middle1(); pre(4);

printf("Item Counting ");

printf(" %d ",cs);

Sleep(150);

cls();

}

cls();

middle1();pre(4);

printf("Total Food Item is --> %d \n",citem); Sleep(2000);

goto adminchoise;

}

///Backup System

else if(adminchoise==7){

char date[35]=\_\_DATE\_\_;

strcat(date,".txt");

FILE \*fptr;

fptr=fopen(date,"w");

backuploader();

if(fptr==NULL){

br(3); pre(3); printf("Error!"); Sleep(500);

goto adminchoise;

}

fprintf(fptr,"Total Cash Today : %0.2f\n\n\n",totalmoney);

fprintf(fptr,"Card No ------- Money \n\n");

for(int l=1; l<=c;l++){

fprintf(fptr,"%d ------- %0.2f \n",cardno[l],cardmoney[l]);

}

br(2);pre(4); printf("Backup Successfull..."); Sleep(1500);

fclose(fptr);

goto adminchoise;

}

else if(adminchoise==8){

cls();br(2);pre(2);

//ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "); //ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("| Order No. | FooD Name | Quantity | In Stock |"); //ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("------------------------------------------------------"); //ccolor(26);

for(int o=1;o<=order;o++){

order\_view(total\_order[o],order\_quantity[o],o);

}

br(2);pre(4); printf("1. <-- back \n\n");pre(5);

fflush(stdin); scanf("%d",&any);

goto adminchoise;

}

else if(adminchoise==0){

goto mainmenu;

}

else{

br(2); pre(4); printf("Please Select From List : "); Sleep(500);

goto adminchoise;

}

}

}

else if(main\_menu\_choice ==3){ //TRIE

///////////////////////////

int select,flag=0;

node \*temp;

temp=(node \*)malloc(sizeof(node));

temp = list;

fflush(stdin);

printf("Enter Food to be searched:");

scanf("%[^\n]s",food);

while(temp!=NULL){

insert(root,temp->foodname);

temp=temp->next;

break;

}

if(search(root, food)){

printf("The Food item you Searched is in the menu\n");

printf("To you order you fav. food now[1 for Yes/0 for No]:");

fflush(stdin); scanf("%d",&select);

if(select)

{

flag=1;

}

else{

flag=0;}

}

else{

printf("The Food item you Searched is not in the menu.We'll try adding it to the menu soon'\n");

flag=0;

Sleep(5000);

}

///////////////////////////

if(flag)

{

goto foodlist;

}

else{

goto mainmenu;

}

}

else if(main\_menu\_choice==4){

cls();

middle1(); pre(3); printf("Thank You For Using Our System \n\n\n\n\n\n\n");

Sleep(1000);

exit(1);

}

}

else{

br(2); pre(4); printf("Please Enter Correct Choice"); Sleep(300);

goto mainmenu;

}

int get\_food\_choice;

br(2); pre(3);fflush(stdin); printf("Place Your Order: ") ; scanf("%d",&get\_food\_choice);

if(get\_food\_choice==0){

goto mainmenu;

}

node \*temp;

temp = list ;

while(temp->data != get\_food\_choice){

temp = temp->next;

if(temp==NULL){

br(2); pre(4); echo("Please Choose From List "); br(2); Sleep(1000);

goto foodlist;

}

}

if(get\_food\_choice == temp->data){

fcquantity:

br(2); pre(4); printf("Enter Food Quantity : ");

int fcquantity;

fflush(stdin); scanf("%d",&fcquantity); cls();

if(fcquantity==0){

cls(); middle1();pre(3); printf("Quantity Can not be Zero "); Sleep(2000);

cls();

goto foodlist;

}

else if(fcquantity>temp->quantity){

cls(); middle1();pre(3); printf("Out of Stock ! "); Sleep(2000);

goto foodlist;

}

middle1();pre(4); printf("Choice food %s its price is %0.2f \n\n",temp->foodname,temp->price\*fcquantity);pre(4);

printf("1. Confirm to buy this \n\n");pre(4);

printf("2. Food List ");

confirm:

int confirm;

fflush(stdin); scanf("%d",&confirm);

if(confirm == 1 ){

br(2);pre(4); printf(" 1. Cash ");

br(2);pre(4); printf(" 2. Credit ");

payment:

int payment;

fflush(stdin); scanf("%d",&payment);

if(payment==1){

totalmoney += temp->price\*fcquantity;

order++;

total\_order[order]=get\_food\_choice;

order\_quantity[order]=fcquantity;

uquantity = temp->quantity - fcquantity;

updatefood(get\_food\_choice,uquantity);

cls();middle1();pre(4); printf("===>THANK YOU<===");

br(2);pre(4); printf("Food Ordered Successfully ...");

br(2);pre(4); printf("1. Wanna Buy Another Delicious ? ");

br(2);pre(4); printf("2. Main Menu ");

psmenu:

int ps\_menu;

fflush(stdin); scanf("%d",&ps\_menu);

if(ps\_menu==1){goto foodlist;}

else if(ps\_menu==2){goto mainmenu;}

else{br(2);pre(4);printf("Please Choice from list : "); goto psmenu;}

}

///Credit Card Option

else if(payment==2){

int card\_number[100];

c++;

cls();middle1();pre(4); printf("Enter Your Card No : ");

fflush(stdin); scanf("%d",&card\_number[c]);

cardno[c] = card\_number[c];

int pin;

br(2);pre(2); printf("Enter Your Card Pin [we never save your pin] : ");

fflush(stdin); scanf("%d",&pin);

cardmoney[c] = temp->price\*fcquantity;

totalmoney += temp->price\*fcquantity;

order++;

total\_order[order]=get\_food\_choice;

order\_quantity[order]=fcquantity;

uquantity = temp->quantity - fcquantity;

updatefood(get\_food\_choice,uquantity);

br(2);pre(4); printf("Payment Success...");

br(2);pre(4); printf("1. Wanna Buy Another Delicious ? ");

br(2);pre(4); printf("2. Main Menu ");

psmenu2:

int ps\_menu2;

scanf("%d",&ps\_menu2);

if(ps\_menu2==1){goto foodlist;}

else if(ps\_menu2==2){goto mainmenu;}

else{br(2);pre(4);printf("Please Choice from list : "); goto psmenu2;}

}

else{

br(2);pre(4); printf("Enter Choice from List : ");

goto payment;

}

} ///END Confirm Y/y

else if(confirm == 2){

goto foodlist;

}

else{

br(2);pre(4); printf("Enter Choise from List : ");

goto confirm;

} ///end confirm;

} ///end get food choice if line

else{

br(2);pre(4); echo("Please Choice From List "); br(2); Sleep(300);

goto foodlist;

} ///end get food choice

}

void cls(){

system("cls");

}

void echo(char print[]){

printf("%s",print);

}

void br(int line){

for(int i=0; i<line;i++){

printf("\n");

}

}

void pre(int tab){

for(int i=0; i<tab;i++){

printf("\t");

}

}

void span(int space){

for(int i=0; i<space;i++){

printf(" ");

}

}

void main\_menu(){

cls();

br(5); pre(3); echo("===> 1. Food List"); Sleep(400);

br(2); pre(3); echo("===> 2. Admin Panel"); Sleep(400);

br(2); pre(3); echo("===> 3. Search if your fav. food is in the menu yet"); Sleep(400);

br(2); pre(3); echo("===> 4. Exit"); Sleep(400);

// br(2); pre(3); echo("=> 4. Admin Panel"); Sleep(400);

br(1);

}

void insertend(int data, char foodname[25], int quantity, float price){

node \*temp;

temp=(node \*)malloc(sizeof(node));

temp->data = data;

temp->price = price;

temp-> quantity = quantity;

strcpy(temp->foodname,foodname);

temp->next = NULL;

if(head==NULL){

head = temp;

list = head;

}

else{

while(head->next != NULL){

head = head->next;

}

head->next = temp;

}

}

void insertfirst(int data, char foodname[25], int quantity, float price){

node \*temp;

temp=(node \*)malloc(sizeof(node));

temp->data = data ;

temp->price = price;

strcpy(temp->foodname,foodname);

temp-> quantity = quantity;

temp->next = head;

head = temp;

list = head ;

}

void insertmid(int pos, int data, char foodname[25], int quantity, float price){

node \*temp;

temp=(node \*)malloc(sizeof(node));

temp->data = data;

temp->price = price;

temp-> quantity = quantity;

strcpy(temp->foodname,foodname);

while(head->next->data != pos ){

head = head->next ;

}

temp->next = head->next;

head->next = temp ;

// free(temp);

}

void deletefood(int serial){

node \*temp;

temp=(node \*)malloc(sizeof(node));

temp = list;

if(temp->data != serial){

while(temp->next->data != serial){

temp = temp->next;

}

if(temp->next->data == serial){

temp->next = temp->next->next;

cls();

printf("\n\n\n\n\t\t\tDeleting Item %d ",serial);for(int cs=0;cs<4;cs++){printf(" .");Sleep(400);}

printf("\n\n\n\n\t\t\tDeleted Successfylly \n"); Sleep(500);

}

else{

printf("\n\n\n\n\t\t\tFood Item Not Found\n"); Sleep(500);

}

head = temp ;

}

else{

temp = temp->next;

cls();

printf("\n\n\n\n\t\t\tDeleting Item %d ",serial);for(int cs=0;cs<4;cs++){printf(" .");Sleep(400);}

printf("\n\n\n\n\t\t\tDeleted Successfylly \n"); Sleep(500);

head = temp ;

list=head;

}

}

void updatefood(int udata, int uquantity){

node \*temp;

temp = list;

while(temp->data!=udata){

temp = temp->next;

}

if(temp->data == udata){

temp->quantity = uquantity;

}

}

int countitem(){

node \*temp;

temp = list;

int countitem=0;

if(temp==NULL){

countitem = 0;

}

else{

countitem = 1;

while(temp->next != NULL){

countitem++;

temp = temp->next;

}

}

return countitem;

}

void foodlist(){

//ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ");//ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("| Food No. | FooD Name | Price | In Stock |");//ccolor(26);

printf("\n\t\t"); //ccolor(240);

printf("-------------------------------------------------------");//ccolor(26);

node \*temp;

temp = list;

while(temp != NULL){

//ccolor(26);

printf("\n\t\t"); //ccolor(62);

printf("| %d | %s | %0.2f | %d |",temp->data,temp->foodname, temp->price, temp->quantity);

//ccolor(26);

printf("\n\t\t"); //ccolor(62);

printf("-------------------------------------------------------");

temp = temp->next ;

Sleep(100);

}

//ccolor(26);

// free(temp);

}

void order\_view(int order, int quantity, int or\_no){

//ccolor(26);

node \*temp;

temp = list;

while(temp->data != order){

temp = temp->next;

}

if(temp->data == order){

//ccolor(26);

printf("\n\t\t"); //ccolor(62);

printf("| %d | %s | %d | %d |",or\_no,temp->foodname,quantity,temp->quantity);

//ccolor(26);

printf("\n\t\t"); //ccolor(62);

printf("-------------------------------------------------------");

Sleep(100);

}

//ccolor(26);

}

void insert(struct TrieNode \*root, char key[25])

{

int level;

int length = strlen(key);

int index;

struct TrieNode \*pCrawl = root;

for (level = 0; level < length; level++)

{

index = CHAR\_TO\_INDEX(key[level]);

if (!pCrawl->children[index])

pCrawl->children[index] = getNode();

pCrawl = pCrawl->children[index];

}

// mark last node as leaf

pCrawl->isEndOfWord = true;

}

bool search(struct TrieNode \*root, char key[25])

{

int level;

int length = strlen(key);

int index;

struct TrieNode \*pCrawl = root;

for (level = 0; level < length; level++)

{

index = CHAR\_TO\_INDEX(key[level]);

if (!pCrawl->children[index])

return false;

pCrawl = pCrawl->children[index];

}

return (pCrawl != NULL && pCrawl->isEndOfWord);

}

/\*bool searchfood(const char foodnme[],struct TrieNode \*root)

{

printf("Enter Food to be searched:");

int select;

node \*temp1;

temp1 = list;

while(temp1 != NULL){

insert(root, temp1->foodname);

temp1 = temp1->next ;

}

if(search(root, foodnme)){

printf("The Food item you Searched is in the menu\n");

printf("To you order you fav. food now[1 for Yes/0 for No]");

fflush(stdin); scanf("%d",&select);

if(select)

{

return true;

}

return false;

}

else{

printf("The Food item you Searched is not in the menu.We'll try adding it to the menu soon'\n");

return false;

}

}\*/

/\*void //ccolor(int clr){

HANDLE hConsole;

hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, clr);

}\*/

///HEERE PRINTF STYLE FUNCTION

void pwellcome(){

//ccolor(26);

char welcome[50]="WELCOME";

char welcome2[50]=" TO";

char welcome3[50]=" FOOD ORDER";

char welcome4[50]=" MANAGEMENT SYSTEM";

printf("\n\n\n\n\n\t\t\t");

for(int wlc=0; wlc<strlen(welcome);wlc++){

//ccolor(120+(wlc\*9));

printf(" %c",welcome[wlc]);

Sleep(200);

}

//ccolor(26);

printf("\n\n\t\t\t\t ");

for(int wlc2=0; wlc2<strlen(welcome2) ;wlc2++){

//ccolor(160+(wlc2\*9));

printf(" %c",welcome2[wlc2]);

Sleep(200);

}

//ccolor(26);

printf("\n\n\n\t\t\t ");

for(int wlc3=0; wlc3<strlen(welcome3) ;wlc3++){

if(welcome3[wlc3]!='D'){

//ccolor(121+(wlc3\*4));

printf(" %c",welcome3[wlc3]);

}

else{

//ccolor(11);

printf(" %c",welcome3[wlc3]);

}

Sleep(200);

}

//ccolor(26);

printf("\n\n\n\t\t\t\t ");

for(int wlc3=0; wlc3<strlen(welcome4) ;wlc3++){

if(welcome4[wlc3]!='A' && welcome4[wlc3]!='E'){

//ccolor(121+(wlc3\*4));

printf(" %c",welcome4[wlc3]);

}

else{

//ccolor(11);

printf(" %c",welcome4[wlc3]);

}

Sleep(200);

}

//ccolor(26);

}

void loadingbar(void){

for (int i=15;i<=100;i+=5){

cls();

//ccolor(26);

printf("\n\n\n\n\n\n\n\t\t\t\t");

printf("%d %% Loading...\n\n\t\t",i);

printf("");

for (int j=0; j<i;j+=2){

//ccolor(160+j);

printf(" ");

//ccolor(26);

}

Sleep(100);

if(i==90 || i==50 || i==96 || i==83){

Sleep(100);

}

}

}

void backuploader(void){

for (int i=15;i<=100;i+=5){

cls();

//ccolor(26);

printf("\n\n\n\n\n\n\n\t\t\t\t");

printf("%d %% Backing UP DATA...\n\n\t\t",i);

printf("");

for (int j=0; j<i;j+=2){

//ccolor(120+j);

printf(" ");

//ccolor(26);

}

Sleep(50);

if(i==90 || i==50 || i==96 || i==83){

Sleep(50);

}

}

}

void middle1(void){

printf("\n\n\n\n\n\n\n");

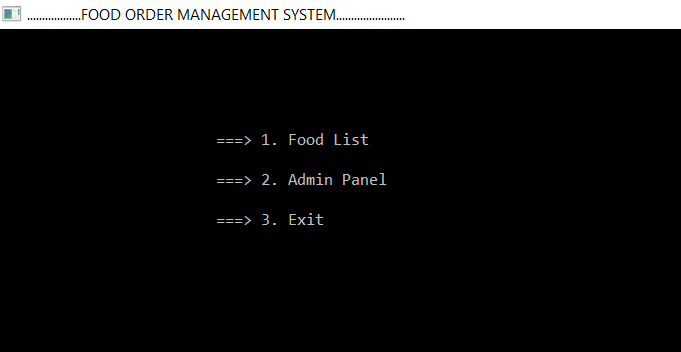
}

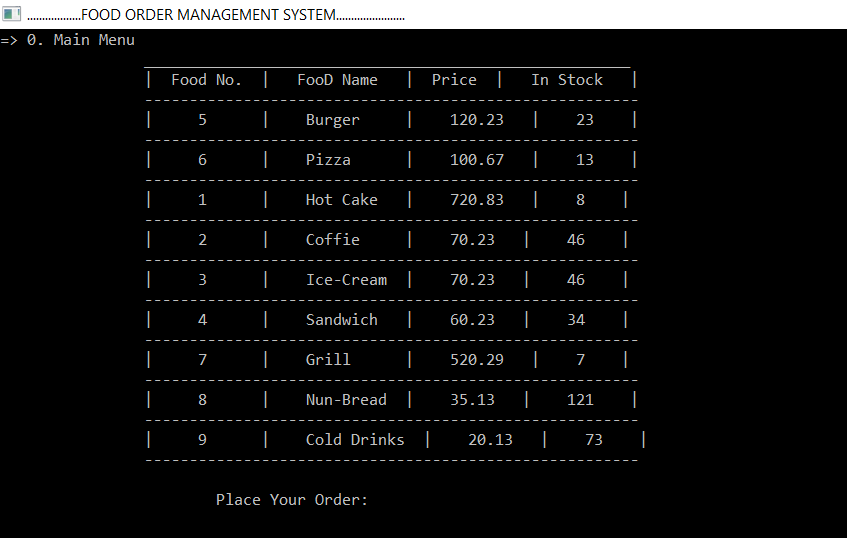
void middtab1(void){

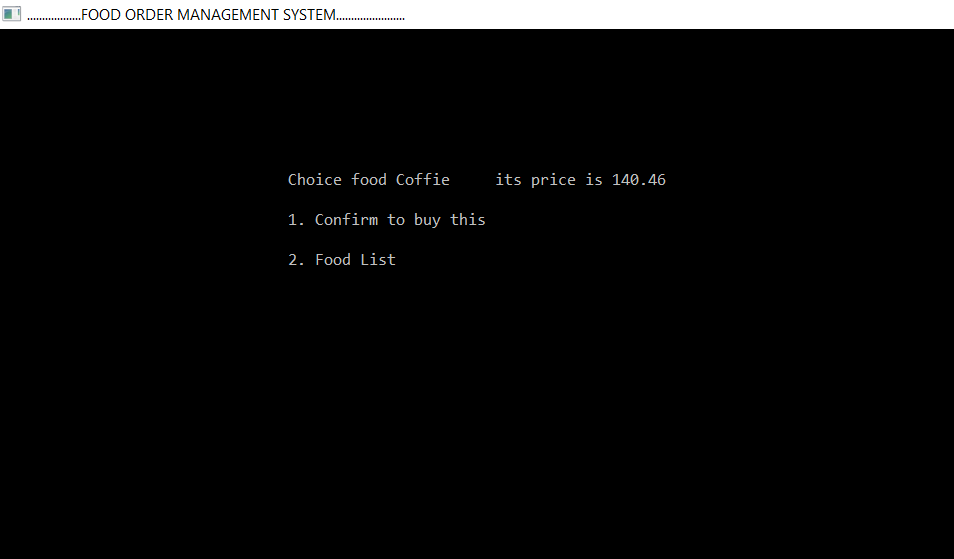
printf("\t\t\t\t\t");

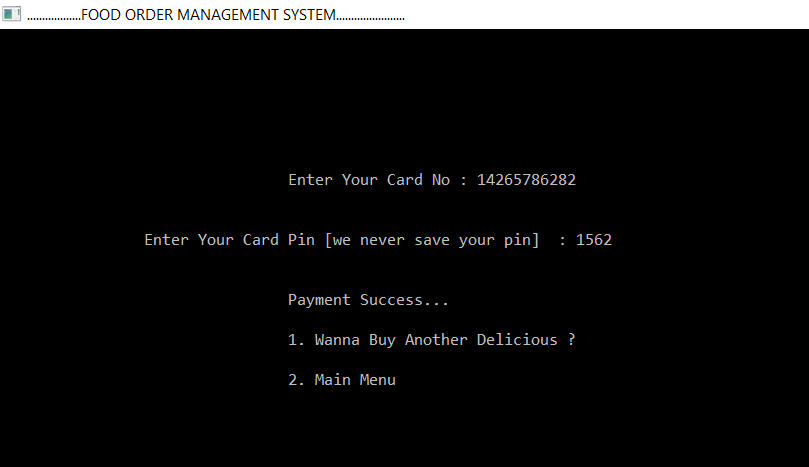
}

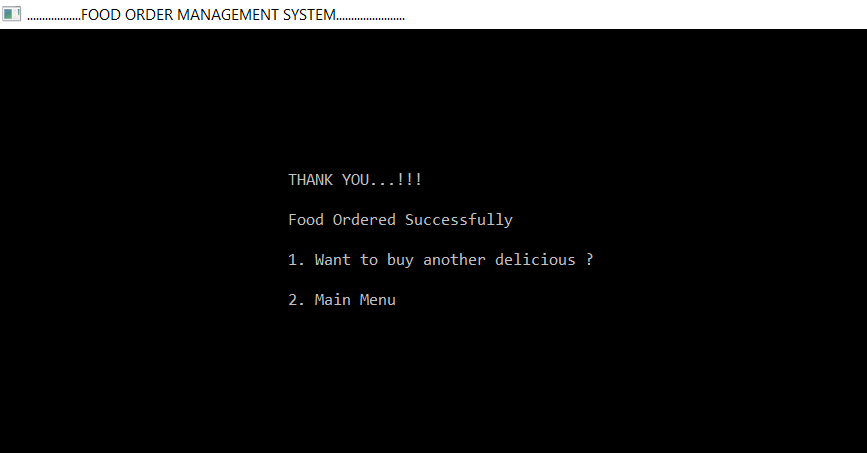
**PROTOTYPE DISPLAY**

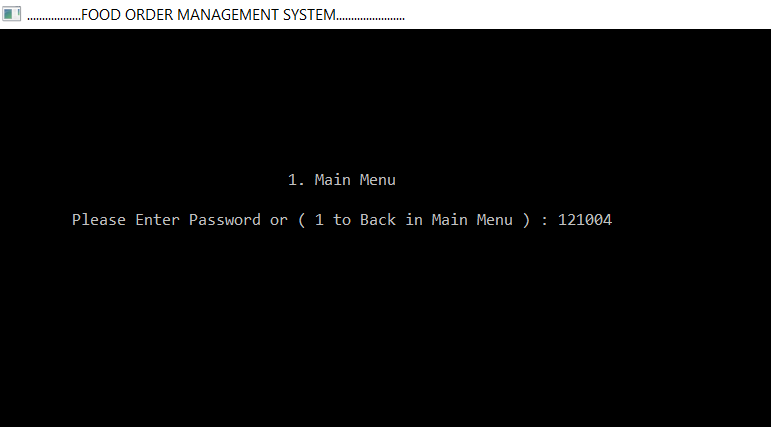
****

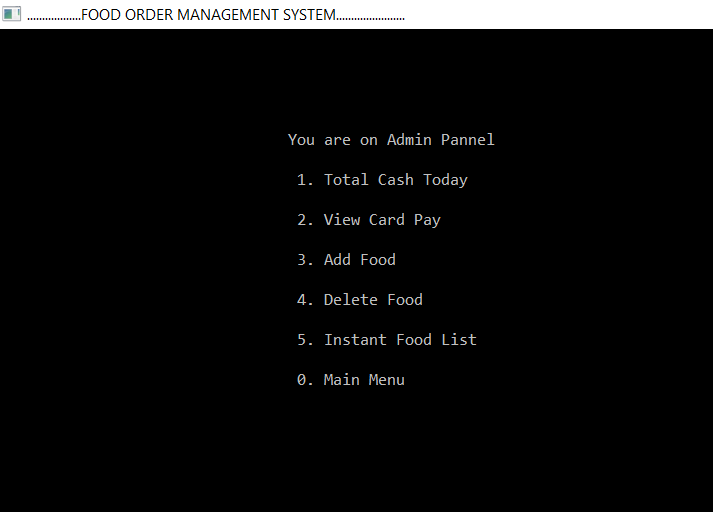
****

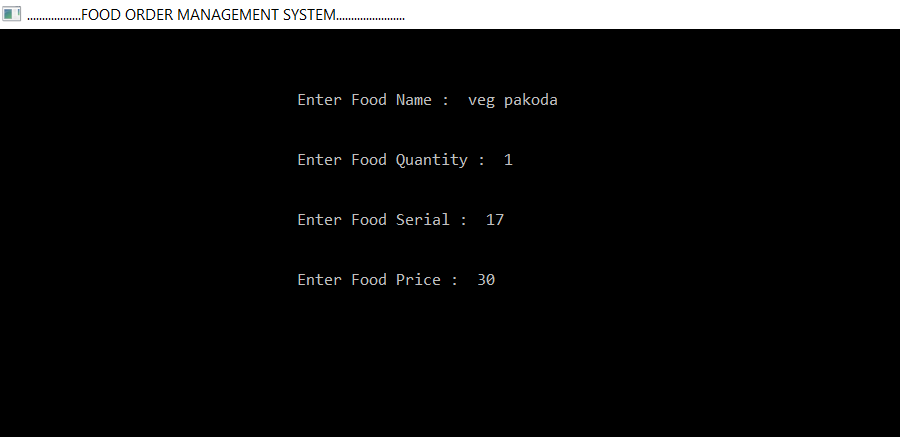
****

****

****

****

****

****

**ADDITIONAL WORK**

1. We can use LCS i.e. dynamic programming for finding the food items more accurately. LCS will help the user to find food items effectively if he/she have typed a small part of the word.
2. We can also make a recommendation system using machine learning. That system would recommend user for their food of interests if he/she visits the restaurant again.

**CONCLUSION**

* This food order management system uses 3 data structures:
* 1. Linked list
* 2. Trie
* 3.Searching
* Hence, customers can easily have the access to their favorite food item , they can see the status of the item available in the food list. Hence this system is flexible and reliable to all the customers.

**REFERENCES**

1. <https://www.geeksforgeeks.org/trie-insert-and-search/>
2. <https://www.geeksforgeeks.org/linked-list-set-1-introduction/>
3. <https://en.wikipedia.org/wiki/Trie>
4. <https://medium.com/basecs/trying-to-understand-tries-3ec6bede0014>