# **FOOD ORDER MANAGEMENT SYSTEM**

*A*

*Mini Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

ANKEM REVATHI -1602-19-737-155

KANDHULA AKHILA -1602-19-737-125



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahim Bagh, Hyderabad-31**

**2020**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Hyderabad-500 031**

**Department of Information Technology**



**DECLARATION BY THE CANDIDATE**

We**, ANKEM REVATHI** and **KANDHULA AKHILA** bearing hall ticket numbers, **1602-19-737-155** and **1602-19-737-125**, hereby declare that the project report entitled “FOOD ORDER MANAGEMENT SYSTEM” is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**.

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

ANKEM REVATHI

1602-19-737-155

KANDHULA AKHILA

1602-19-737-125

(Faculty In-Charge) (Head, Dept of IT)

**ACKNOWLEDGEMENTS**

The satisfaction that accompanies the successful completion of this project would not be in complete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. We consider ourselves privileged to express gratitude and respect towards all those who guided us through the completion of this project.

We convey thanks to my project guide Mrs.Divya of Information technology Department for providing encouragement, constant support and guidance which was of a great help to complete this project successfully.

Last but not the least, we wish to thank our parents for financing our studies in this privileged Vasavi College of Engineering as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

**ABSTRACT**

Food Order Management System is based on the concept of ordering food and managing employee records. There’s no login system available for this system, the user can freely use its feature. This application has two categories i.e., Admin section and customer section. Admin section, from where the user can manage employee and customer section, from where a user can order food. The user can simply order foods by selecting food items, entering a customer name, quantity. From the admin section, the user can manage employee records by entering his/her name, phone number, address, department, date of birth, and joining date. The admin can also view the last orders of the customers.

**Table of contents**

|  |  |
| --- | --- |
| **1.0 Introduction** | 1 |
| **2.0 Technology**  a) software requirements  b) hardware requirements | 2 |
| **3.0 Proposed work**  **3.1** **DESIGN**  3.11 Use case diagram and description for all the use cases……..  **3.2 IMPLEMENTATION**  3.21 Module wise code for the entire project….  3.22 GitHub links  **3.3 TESTING** | 3-5  5-35  35  35-41 |
| **4.0 Results** | 42-60 |
| **5.0 Additional knowledge gained** | 60 |
| **6.0 Conclusion and Future Work** | 61 |
| **7.0 References** | 61 |

**INTRODUCTION**

Food ordering is an application which will help restaurant to optimised and control over their restaurants. For the waiters, it is making life easier because they don’t have to go kitchen and give the orders to the chef easily.

For the management point of view, the admin will be able to control the administration block by having all the reports to hand and able to see the records of each employees and orders.

This application helps the restaurant to do all functionalities more accurately and faster way. Food ordering system reduces manual works and improves efficiency of restaurant. This application is helping food orderings to maintain the stock and cash flows and there are many more functionalities like,

* To store records
* Control orders and services
* Billings
* Control staff and their records
* Helps admin to control each part of administration

**TECHNOLOGY**

All compute software needs certain hardware components or other software components resources to be present. In order for computers to used efficiently these are the primary requisites. There are two different categories of this section.

1. **Software Requirements:**

Software requirements mainly share out with defining the software resource requirements that need to be installed on a computer to provide optimal functioning of a particular application. These conditions are not included in the software installation package and need to be installed separately.

In order to use Parking Lot Management System, the following are the prerequisites,

1. Operating System – Windows 7 and above
2. C Compiler – GNU Compiler
3. Editor – Any basic editor is preferable (Example: Notepad++)

1. **Hardware Requirements**:

Hardware requirements refer to the common set requirements defined by any system or software application and are usually the physical computer resources. In this section we basically deal with primary memory, secondary memory, processing power.

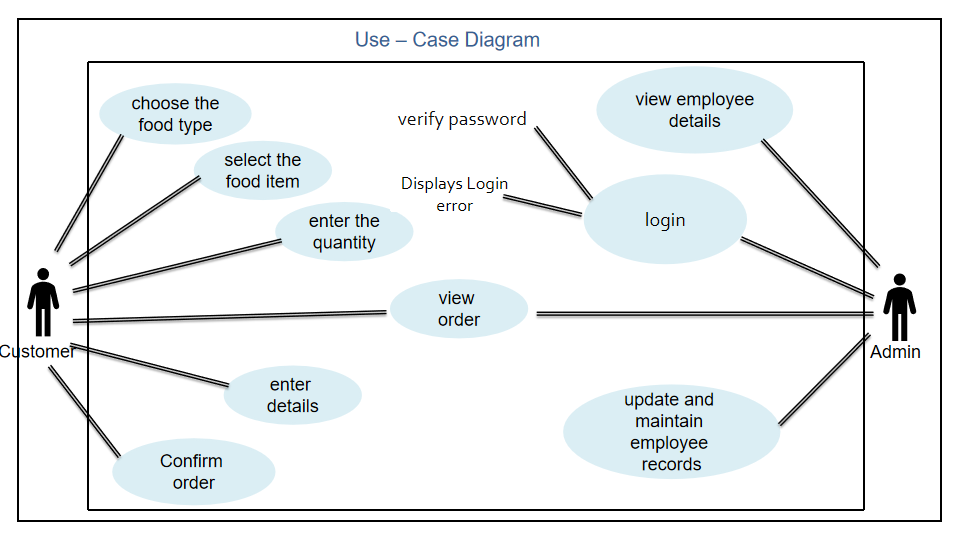
In order to use Parking Lot Management System, one should have the following,

1. Processor - Intel Core i5 and above
2. Memory – 4GB Ram

**PROPOSED WORK**

1. **DESIGN**

**i) Use case diagram and descriptions for all the use-cases**



**DESCRIPTION OF USE CASES**

* **Choose the food type**

Customer should choose food type(veg/non-veg). According, to customer choice system, will be displayed the menu along with price.

* **Select the food type item**

Now customer can choose their choice of food items

* **Enter the quantity**

After choosing the food items system, will ask how much quantity do customer need After choosing the quantity system asks whether customer need any add ups or to display the total price. Now according to customer choice if customer needs any add ups then menu will display according or otherwise total price will be shown.

* **Enter the details**

After displaying the price details of the customer should be filled and then customer details will be saved.

* **Confirm order**

Now customer can conform their order system will display total price and details of the customer along with appropriate message.

* **Admin Password**

Admin needs to enter the password to manage the employee details. If Admin enters correct password admin goes to employee section or else admin remains still, until admin enters the correct password.

* **Add an Employee details**

After reaching the employees section if admin wants to add the employee details admin need to choose add an employee details option. Then admin needs enter employee details such as Name, Id, date of birth, year of joining, department, address. Then system asks the admin whether admin wants to add any other details of employee or not. If admin wants to enter details of another employee then admin can choose yes or else no. If admin choose Yes then admin needs to enter another employee details this process will continue until the admin wants to exit. If admin choose No then employee section will display.

* **Search an Employee details**

If admin wants to refer to details of the employee then he can search with employee Id or name. If admin enters Name if the Name is found in the records the then the details of respective employee are displayed else terminated. If admin enters Id if the Id is found in the records the then the details of respective employee are displayed else terminated.

* **Delete an Employee details**

If admin wants to delete any employee details then he needs to enter the employee id then the details of respective employee are displayed then system will ask the admin to delete this employee details or not then admin can choose delete and the message is displayed that the details are deleted successfully. Then system asks the admin whether admin wants to delete any other details of employee or not

If admin wants to delete details of another employee then admin can choose yes or else no. If admin choose Yes then admin needs to delete another employee details these process will continue until the admin wants to exit. If admin choose No then employee section will display.

* **View the employee records**

Admin will be redirected to employee section if admin wants to view all the details of every employee then he can choose view option. All the details of every employee are displayed.

* **Shows the previous order details**

If admin want to see the previous order history then admin should choose the previous history option then total price, name, and address of each individual customers will be displayed and admin can view the total history of orders till the date.

1. **Implementation:**

**i) Module wise code for the entire project**

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

**//FUNCTIONS**

void returnfunc(void);

void mainmenu(void);

void administration(void);

void customer(void);

void addrecord(void);

void deleterecord(void);

void searchrecord(void);

void viewrecord(void);

void vegetarian(void);

void nonvegetarian(void);

void details(void);

void Password(void);

int getdata();

struct employee

{

char name[30];

int id;

int dd; int mm; int yyyy;

int YOJ;

char place[20];

float salary;

char department[20];

int quantity;

};

struct employee e;

FILE \*fp,\*ft,\*dp,\*dt;

int s;

char findrecord;

int again;

int quantity;

double total=0;

int t;

char password[20]="foodgood";

int main()

{

mainmenu();

return 0;

}

//**FUNCTION NAME : void mainmenu(void)**

void mainmenu(void)

{

system("cls");

printf("\n\n");

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

printf(" -----------------------WELCOME TO FOOD ORDER SYSTEM-----------------------\n");

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

printf(" 1. ADMINISTRATION SECTION--> \n\n 2. CUSTOMER SECTION--> \n\n 3. Exit--> \n\n\n Enter Your Choice:");

int choice;

scanf("%d",&choice);

if(choice==1)

{

administration();

}

else if(choice==2)

{

customer();

}

else if(choice==3)

{

system("cls");

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

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*-------------------THANK YOU------------------\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

exit(0);

}

}

//**FUNCTION NAME : void administration(void)**

void administration(void)

{

Password();

system("cls");

printf("\n\n");

printf("-----------------------------WELCOME TO THE EMPLOYEE SECTION----------------------------\n\n");

printf(" 1. Add New Record-->\n\n 2. Search Record-->\n\n 3. Delete Record-->\n\n 4. View THe Records-->\n\n 5. View previous orders placed-->\n\n 6. Back To Main Menu-->\n\n Enter Your Choice:");

int n;

scanf("%d",&n);

if(n==1)

{

addrecord();

}

else if(n==2)

{

searchrecord();

}

else if(n==3)

{

deleterecord();

}

else if(n==4)

{

viewrecord();

}

else if(n==5)

{

FILE \*view;

char viw;

view = fopen("order.txt","rt");

while((viw=fgetc(view))!=EOF)

{

printf("%c",viw);

}

fclose(view);

mainmenu();

}

else if(n==6)

{

system("cls");

mainmenu();

}

else

{

printf("Wrong Input !! PLease Re-enter The Correct Option");

if(getch())

administration();

}

}

//**FUNCTION NAME : void addrecord(void)**

void addrecord(void)

{

system("cls");

fp=fopen("record1.txt","a");

if(getdata()==1)

{

fseek(fp,0,SEEK\_END);

fwrite(&e,sizeof(e),1,fp);

fclose(fp);

printf("\n\n");

printf("The Record Is Successfully Saved ! !\n\n");

printf("Save any more?(y / n): ");

if(getch()=='n')

administration();

else

system("cls");

addrecord();

}

}

//**FUNCTION NAME : void deleterecord(void)**

void deleterecord(void)

{

system("cls");

int d;

char another='y';

while(another=='y') //infinite loop

{

system("cls");

printf(" Enter The Employee ID To Delete :");

scanf(" %d",&d);

fp=fopen("record1.txt","r+");

rewind(fp);

while(fread(&e,sizeof(e),1,fp)==1)

{

if(e.id==d)

{

printf("\n\n");

printf(" .....................The Employee Record Is Available....................\n\n");

printf(" Employee Name Is %s\n\n",e.name);

printf(" Employee's Date OF Birth Is %d/%d/%d\n\n",e.dd,e.mm,e.yyyy);

findrecord='t';

}

}

if(findrecord!='t')

{

printf(" .........................No record is found modify the search..........................\n\n");

if(getch())

administration();

}

if(findrecord=='t' )

{

printf("Do You Want To Delete THe Record ? (y / n) ");

if(getch()=='y')

{

ft=fopen("test1.txt","w"); //temporary file for delete

rewind(fp);

while(fread(&e,sizeof(e),1,fp)==1)

{

if(e.id!=d)

{

fseek(ft,0,SEEK\_CUR);

fwrite(&e,sizeof(e),1,ft); //write all in tempory file except that

} //we want to delete

}

fclose(ft);

fclose(fp);

remove("record1.txt");

rename("test1.txt","record1.txt"); //copy all item from temporary file to fp except that

fp=fopen("record1.txt","r"); //we want to delete

if(findrecord=='t')

{

printf(" THE RECORD IS SUCCESSFULLY DELETED.\n\n");

printf(" Delete Another Record ? (Y/N) : ");

}

}

else

administration();

fflush(stdin);

another=getch();

}

}

administration();

}

//**FUNCTION NAME : void searchrecord(void)**

void searchrecord(void)

{

system("cls");

int d;

printf(" -------------------------Search Employees Record-------------------------\n\n");

printf(" 1. Search By ID\n");

printf(" 2. Search By Name\n\n");

printf(" Enter Your Choice---> ");

fp=fopen("record1.txt","r"); //open file for reading propose

rewind(fp); //move pointer at the begining of file

switch(getch())

{

case '1':

{

system("cls");

printf(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Search Record By Id\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" Enter The Employee ID : ");

scanf(" %d",&d);

printf("\n\n");

printf("Searching........");

while(fread(&e,sizeof(e),1,fp)==1)

{

if(e.id==d)

{

printf("\n\n");

printf("...........................The Record is available............................\n\n");

printf(" ID : %d\n\n",e.id);

printf(" Name : %s\n\n",e.name);

printf(" Date OF Birth :%d/%d/%d\n\n",e.dd,e.mm,e.yyyy);

printf(" Year Of Joining :%d\n\n",e.YOJ);

printf(" Department :%s\n\n",e.department);

printf(" Place :%s\n\n",e.place);

findrecord='t';

}

}

if(findrecord!='t') //checks whether condition enters inside loop or not

{

printf("\aNo Record Found\a");

}

printf("Try Another Search ? (y/n)");

if(getch()=='y')

searchrecord();

else

administration();

break;

}

case '2':

{

char s[15];

system("cls");

printf(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Search Employees By Name\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" Enter Employee Name : ");

scanf(" %s",s);

int d=0;

while(fread(&e,sizeof(e),1,fp)==1)

{

if(strcmp(e.name,(s))==0) //checks whether a.name is equal to s or not

{

printf(" .........................The Record is available...........................\n\n\n");

printf(" ID : %d\n\n",e.id);

printf(" Name : %s\n\n",e.name);

printf(" Date OF Birth :%d/%d/%d\n\n",e.dd,e.mm,e.yyyy);

printf(" Year Of Joining :%d\n\n",e.YOJ);

printf(" Department :%s\n\n",e.department);

printf(" Place :%s\n\n",e.place);

d++;

}

}

if(d==0)

{

printf("\aNo Record Found\a");

}

printf("Try Another Search ? (Y/N)");

if(getch()=='y')

searchrecord();

else

administration();

break;

}

default :

getch();

searchrecord();

}

fclose(fp);

}

//**FUNCTION NAME : void viewrecord(void)**

void viewrecord(void)

{

system("cls");

printf(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Employee Details\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" DEPARTMENT ID EMPLOYEE NAME D.O.B Y.O.J PLACE\n\n");

fp=fopen("record1.txt","r");

while(fread(&e,sizeof(e),1,fp)==1)

{

printf(" %s\t",e.department);

printf("%d\t",e.id);

printf("%s\t",e.name);

printf("%d/%d/%d\t",e.dd,e.mm,e.yyyy);

printf("%d\t",e.YOJ);

printf("%s\t",e.place);

printf("\n\n");

}

fclose(fp);

returnfunc();

}

void returnfunc(void)

{

{

printf(" Press ENTER to return to main menu");

}

e:

if(getch()==13) //allow only use of enter

administration();

else

goto e;

}

//**FUNCTION NAME : int checked(int c)**

int checkid(int c) //check whether the record is exist in list or not

{

rewind(fp);

while(fread(&e,sizeof(e),1,fp)==1)

if(e.id==c)

return 0; //returns 0 if employee exits

return 1; //return 1 if it not

}

//**FUNCTION NAME : int getdata()**

int getdata()

{

//\*int t;

printf(" Employee ID :\t");

scanf("%d",&t);

if(checkid(t)==0)

{

printf("\n\n");

printf("\a The Employee Record Already Exists !!!\a");

getch();

mainmenu();

return 0;

}

e.id=t;

printf(" Employee Name : ");

scanf("%s",e.name);

printf(" Date Of Birth (dd/mm/yyyy) :");

scanf("%d/%d/%d",&e.dd,&e.mm,&e.yyyy);

printf(" Year OF Joining :");

scanf("%d",&e.YOJ);

printf(" Place :");

scanf("%s",e.place);

printf(" Department :");

scanf("%s",e.department);

return 1;

}

//**FUNCTION NAME : void customer(void)**

void customer(void)

{

system("cls");

printf("\n\n");

printf(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n ---------------------WELCOME--------------------\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" YOU ARE--->\n\n 1. Vegetarian-->\n\n 2. Non Vegetarian-->\n\n OR\n\n 3. Back To Main Menu-->\n\n Enter Your Choice:");

int k;

scanf("%d",&k);

if(k==1)

{

system("cls");

vegetarian();

}

else if(k==2)

{

system("cls");

nonvegetarian();

}

else if(k==3)

{

system("cls");

mainmenu();

}

else

{

printf("Wrong Input !Please Re-enter The Correct Option......\n\n");

customer();

}

}

//**FUNCTION NAME : void vegetarian(void)**

void vegetarian(void)

{

int choice;

int again;

int quantity;

double price=0;

printf("\n\n");

printf(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n ---------------------VEGETARIAN MENU------------------\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf(" 1. Burger 40/-\n\n 2. Noodles 60/-\n\n 3. Manchuria 45/-\n\n 4. Beverages 35/-\n\n 5. Veg. Thaali 150/-\n\n 6. Sandwich 50/-\n\n 7. Veg lollipop 70/-\n\n 8. Back To Main Menu\n\n Enter Your Choice--> ");

char v;

FILE \*vmenu;

vmenu=fopen("vegmenu.txt","rt");

while((v=getc(vmenu))!=EOF)

{

printf("%c",v);

}

fclose(vmenu);

printf("\n\nEnter What You Want:");

printf("\n\n");

scanf("%d",&choice);

if(choice==1)

{

printf("Quantity : ");

scanf("%d",&quantity);

total=total + 40\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

printf("\n");

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==2)

{

printf("Quantity :");

scanf("%d",&quantity);

total=total + 60\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==3)

{

printf("Quantity : ");

scanf("%d",&quantity);

total=total + 45\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==4)

{

printf("Quantity :");

scanf("%d",&quantity);

total=total + 35\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==5)

{

printf("Quantity :");

scanf("%d",&quantity);

total=total + 150\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==6)

{

printf("Quantity :");

scanf("%d",&quantity);

total=total + 50\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==7)

{

printf("Quantity :");

scanf("%d",&quantity);

total=total + 70\*(quantity);

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf("%d",&again);

if(again==1)

{

printf("\n\n");

vegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is : Rs. %.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==8)

{

system("cls");

customer();

}

else

{

printf("Wrong Input! Re-enter The Correct Option\n\n");

if(getch())

vegetarian();

}

}

//**FUNCTION NAME : void nonvegetarian(void)**

void nonvegetarian(void)

{

int choice;

int again;

int quantity;

double price=0;

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

printf(" ----------------------Non-Vegetarian Menu-----------------\n");

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

printf("1. Chilli chicken 100/-\n\n 2. Chicken 65 120/-\n\n 3. Chicken Biryani 85/-\n\n 4. Beverages 35/-\n\n 5. Mutton kurma 175/-\n\n 6. fish fry 200/-\n\n 7. Back To Main Menu\n\n\nEnter Your Choice");

char n;

FILE \*nmenu;

nmenu = fopen("nonveg.txt","rt");

while((n=getc(nmenu))!=EOF)

{

printf("%c",n);

}

printf("\n\nEnter What You Want :");

printf("\n\n");

scanf("%d",&choice);

if(choice==1)

{

printf("Quantity : ");

scanf(" %d",&quantity);

total=total + 100\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==2)

{

printf("Quantity :");

scanf(" %d",&quantity);

total=total + 120\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==3)

{

printf("Quantity :");

scanf(" %d",&quantity);

total=total + 85\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==4)

{

printf("Enter Quantity :");

scanf(" %d",&quantity);

total=total + 35\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==5)

{

printf("Enter Quantity :");

scanf(" %d",&quantity);

total=total + 170\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==6)

{

printf("Enter Quantity :");

scanf(" %d",&quantity);

total=total + 200\*(quantity);

printf("\n");

printf("Press 1 To Order Again :\nPress 2 To Get Your Total:\n\nEnter Your Choice :");

scanf(" %d",&again);

if(again==1)

{

printf("\n\n");

nonvegetarian();

}

else if(again==2)

{

printf("Your Total Amount Is :%.2lf\n\n",total);

int totl = total;

FILE \*ttl;

ttl = fopen("order.txt","a");

fprintf(ttl,"\nGrand Total : %d\n",totl);

fclose(ttl);

details();

}

}

else if(choice==7)

{

system("cls");

customer();

}

else

{

printf("Wrong Input! Re-enter The Correct Option\n\n");

nonvegetarian();

}

}

//**FUNCTION NAME : void details(void)**

void details(void)

{

char name1[20];

char name2[20];

long long int phone;

char address[40];

char landmark[30];

printf("Please Give Your Contact Details \n");

printf(" First Name : ");

scanf("%s",name1);

printf("Last Name : ");

scanf("%s",name2);

printf("Phone : ");

scanf("%lld",&phone);

printf("Address : ");

scanf("%s",address);

printf("Landmark : ");

scanf("%s",landmark);

printf("\n\n");

printf("Your Entered Details Are --->\n");

FILE \*cust;

cust = fopen("order.txt","a");

fprintf(cust,"Order Placed By : %s %s\nPhone number : %lld\n",name1,name2,phone);

fclose(cust);

printf(" -->First Name : %s\n -->Last Name : %s\n -->Phone :%lld \n -->Address :%s \n -->Landmark :%s \n",name1,name2,phone,address,landmark);

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

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

printf(" ------------------------Your Order Will Be At Your Door In 30 minutes-------------------------\n");

printf(" .....HAPPY ORDERING.....\n");

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

printf("Press Any key To The Main Menu.");

if(getch())

customer();

}

//**FUNCTION NAME : void Password(void)**

void Password(void)

{

system("cls");

char d[30]="PASSWORD PROTECTED";

char ch,pass[10];

int i=0,j;

for(j=0;j<20;j++)

{

printf("\*");

}

for(j=0;j<20;j++)

{

printf("%c",d[j]);

}

for(j=0;j<20;j++)

{

printf("\*");

}

printf("\nEnter Password : ");

while(ch!=13)

{

ch=getch();

if(ch!=13 && ch!=8)

{

putch('\*');

pass[i]=ch;

i++;

}

}

pass[i]='\0';

if(strcmp(pass,password)==0)

{

printf("PASSWORD MATCHED !!!\n\n");

printf("Press Any Key To Continue......\n\n");

getch();

}

else

{

printf("\nWARNING ! INCORRECT PASSWORD....");

printf("\nClick ENTER to enter the password again:");

getch();

Password();

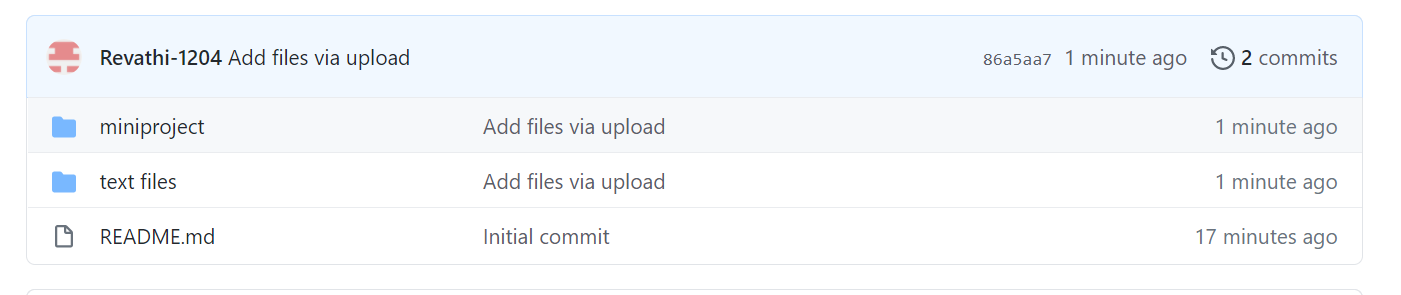
}

}

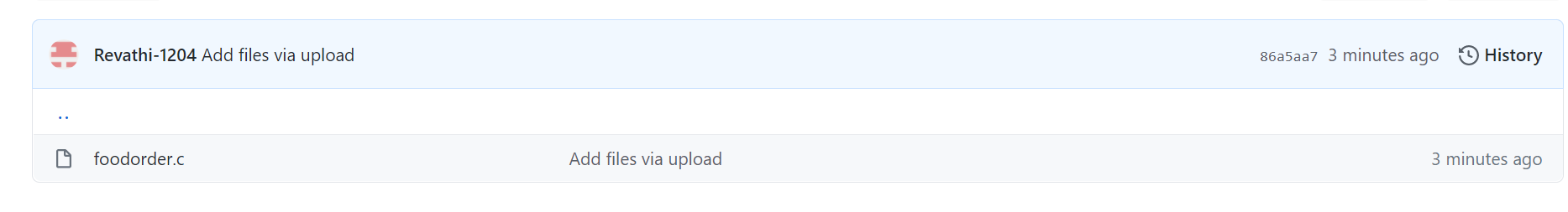
**Git hub links:**

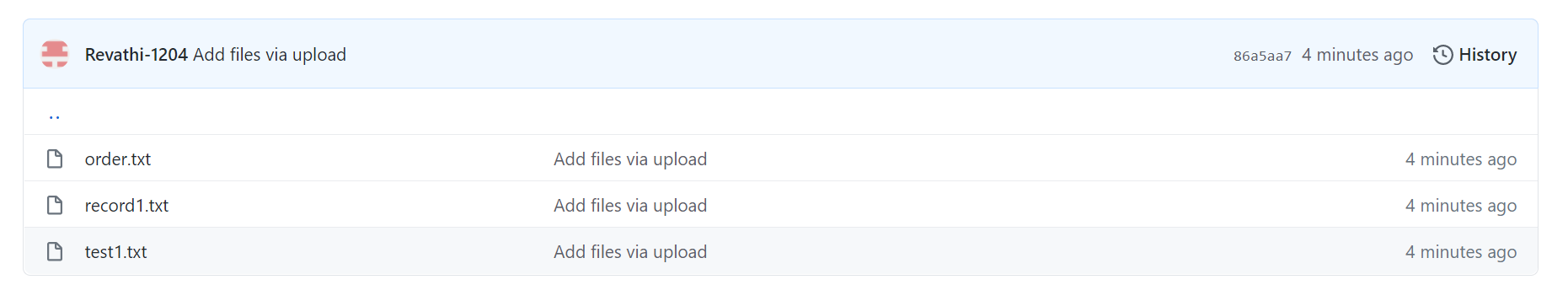
https://github.com/kandulaAkhila

<https://github.com/Revathi-1204>



**Code folder contents**

****

**Files folder contents**

**c)TESTING:**

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-01** |
| **Test case title** : Chooses food type | | |
| **Test case description** : Customer chooses the food type veg/non-veg | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the customer to choose ‘1’ for veg or ‘2’ for non-veg . Customer enters ‘1’. | The menu of selected food type should be displayed. | The veg menu is displayed. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-01** |
| **Test case title** : Chooses food type | | |
| **Test case description** : Customer chooses the food type veg/non-veg | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the customer to choose ‘1’ for veg or ‘2’ for non-veg . Customer enters ‘5’. | The menu of selected food type should be displayed. | Displays “Wrong Input! Please Re-enter The Correct Option”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-02** |
| **Test case title** : Chooses food items. | | |
| **Test case description** : Allows Customer chooses the food items | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Customer enters the ‘Number’ of the selected food item. | The system should ask the quantity of the selected food item | Displays “Enter Quantity” of the selected food item. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-02** |
| **Test case title** : Chooses food items. | | |
| **Test case description** : Allows Customer chooses the food items | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Customer enters an invalid  number. | The system should ask the quantity of the selected food item | Displays “Wrong Input! Re-enter The Correct Option”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-03** |
| **Test case title** : Enters Quantity of food item. | | |
| **Test case description** : Customer enters the quantity. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| After selecting the food item, the system asks the customer to enter the quantity of the food item. The customer enters quantity. | The system should display  Add an item  Get the total amount. | Displays  “Press 1 To Order Again  Press 2 To Get Your Total” | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-03** |
| **Test case title** : Enters Quantity of food item. | | |
| **Test case description** : Customer enters the quantity. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| After selecting the food item, the system asks the customer to enter the quantity of the food item. The customer enters quantity. | The system should display  Add an item  Get the total amount. | “Press 1 To Order Again  Press 2 To Get Your Total” | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-04** |
| **Test case title** : Enters the details. | | |
| **Test case description** : Customer enters the details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| After selecting the food items and quantity of them the system asks the customer to enter the details. | The system should display  The total amount and the details entered by the customer. | Displays bill and the details of customer. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-05** |
| **Test case title** : Confirm order | | |
| **Test case description** : Customer confirms the order. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| After selecting the food items and quantity of them then customer enters wrong details. | The system should display  The total amount and the details entered by the customer. | Displays “Wrong details! please re-enter again”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-06** |
| **Test case title** : login | | |
| **Test case description** : Admin enters the password to login. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the admin to enter password to log into the administration blog. | The system should display password matched and Admin block. | Displays “Password matched” and administration block. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-06** |
| **Test case title** : login | | |
| **Test case description** : Admin enters the password to login. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| The system prompts the admin to enter password to log into the administration blog. Enters wrong password. | The system should display password matched and Admin block. | Displays  “Incorrect Password!  Enter correct password”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-07** |
| **Test case title** : Add an employee details. | | |
| **Test case description** : Allows admin to add an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin adds an employee detail. | The system should verify the details and then updates. | Displays  “The Record Is Successfully Saved”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-07** |
| **Test case title** : Add an employee details. | | |
| **Test case description** : Allows admin to add an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin adds an employee detail that is already in existing. | The system should verify the details and then updates. | Displays  “The Employee Record Already Exists”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-08** |
| **Test case title** : Search an employee details. | | |
| **Test case description** : Allows admin to search an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin searches an employee details either by id or name. | The system should search and display the details. | Displays  “The Record Is Available” and shows the details. | |

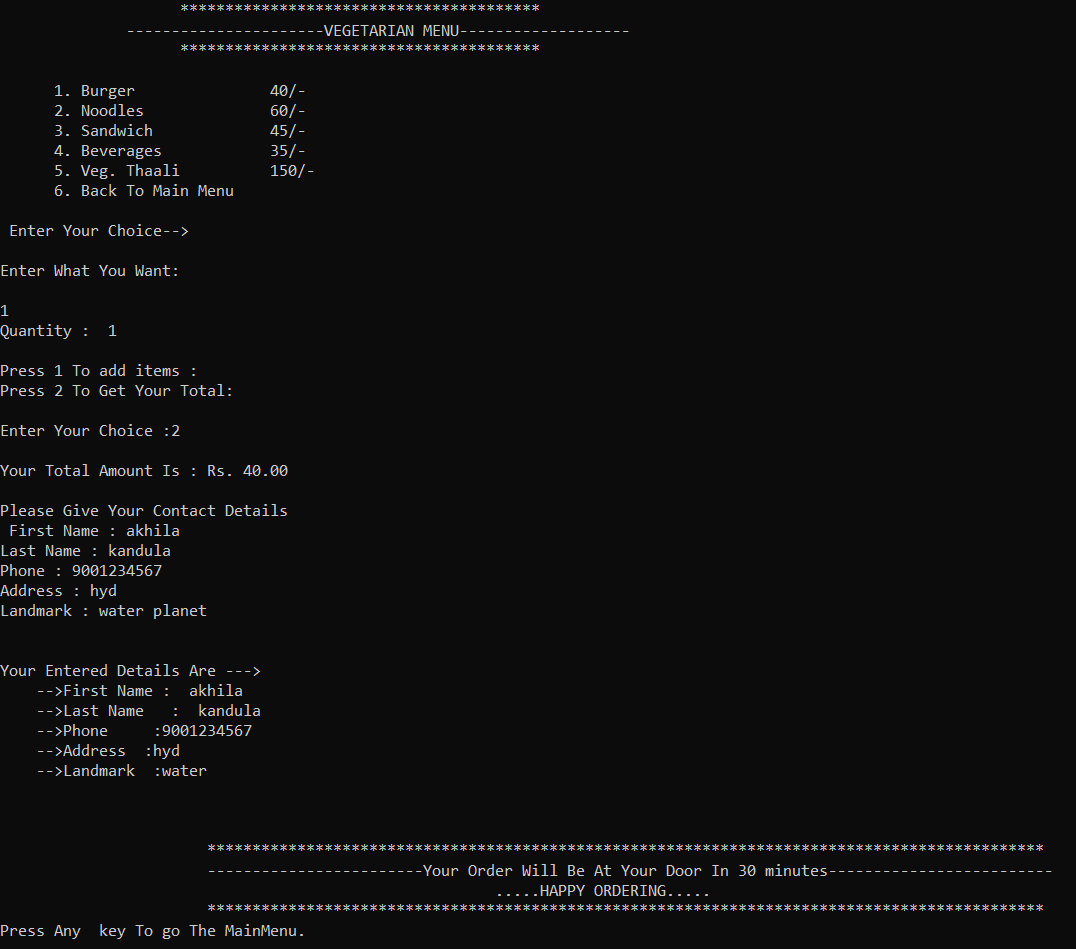
|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-08** |
| **Test case title** : Search an employee details. | | |
| **Test case description** : Allows admin to search an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin searches an employee details either by id or name. | The system should search and display the details. | Displays  “No Record found ” and shows the details. | |

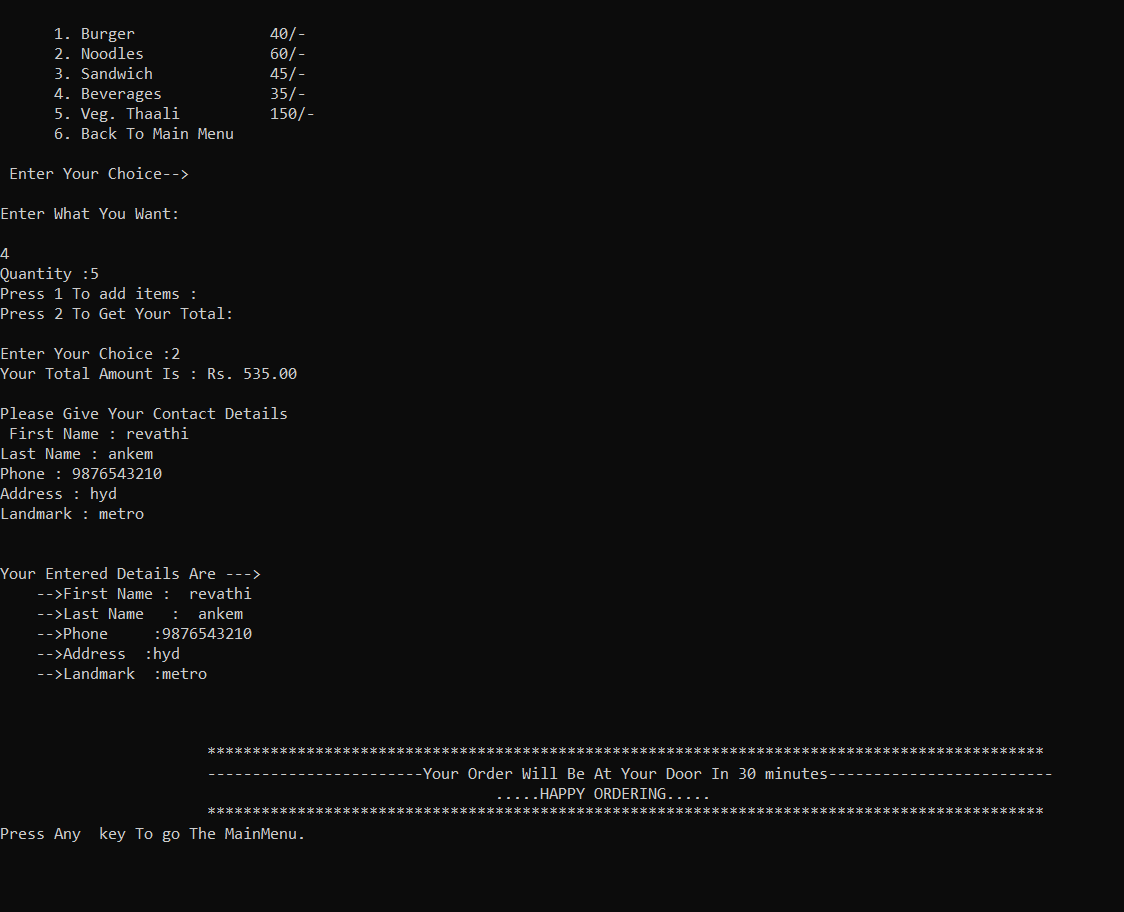
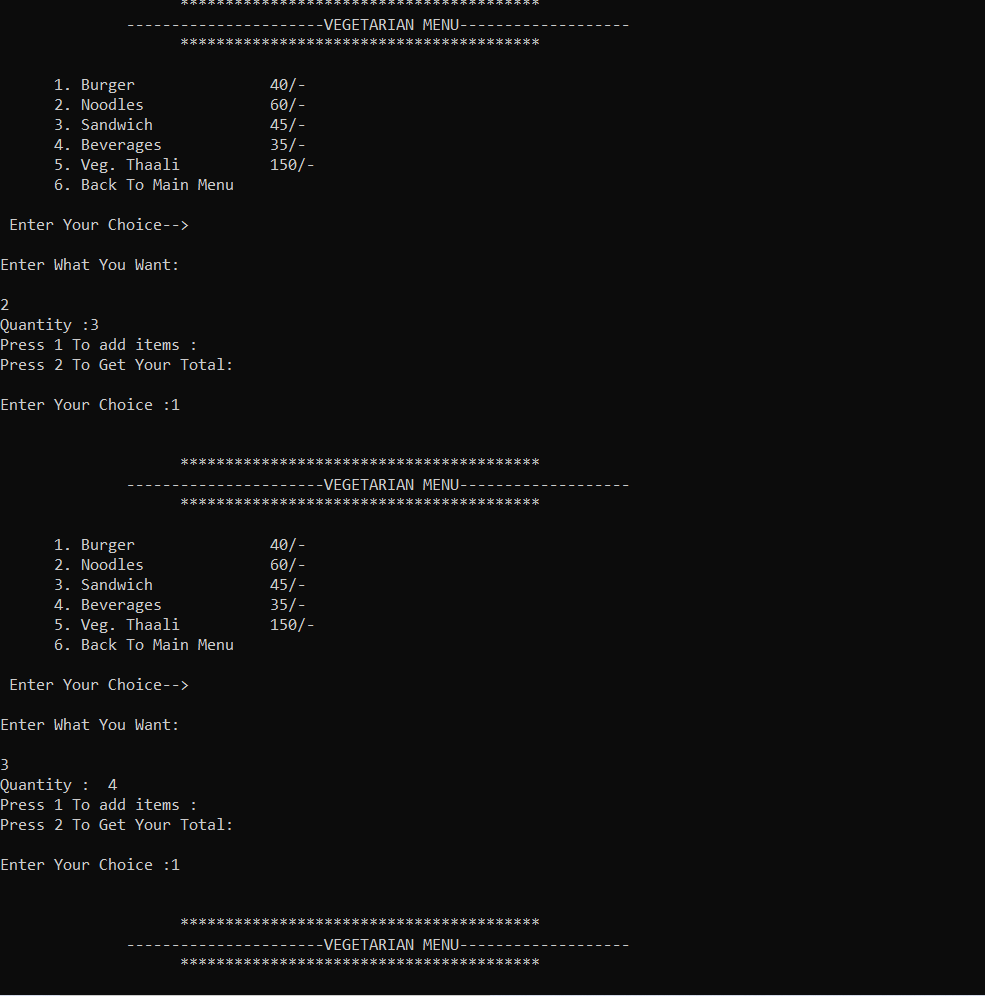
|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-09** |
| **Test case title** : Delete an employee details. | | |
| **Test case description** : Allows admin to delete an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin deletes an employee details by entering id. | The system should first search the employee detail and then deletes. | Displays  “The Record Is Successfully deleted”. | |

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-09** |
| **Test case title** : Delete an employee details. | | |
| **Test case description** : Allows admin to delete an employee details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin deletes an employee details by entering id. | The system should first search the employee detail and then deletes. | Displays  “The Record is not available” . | |

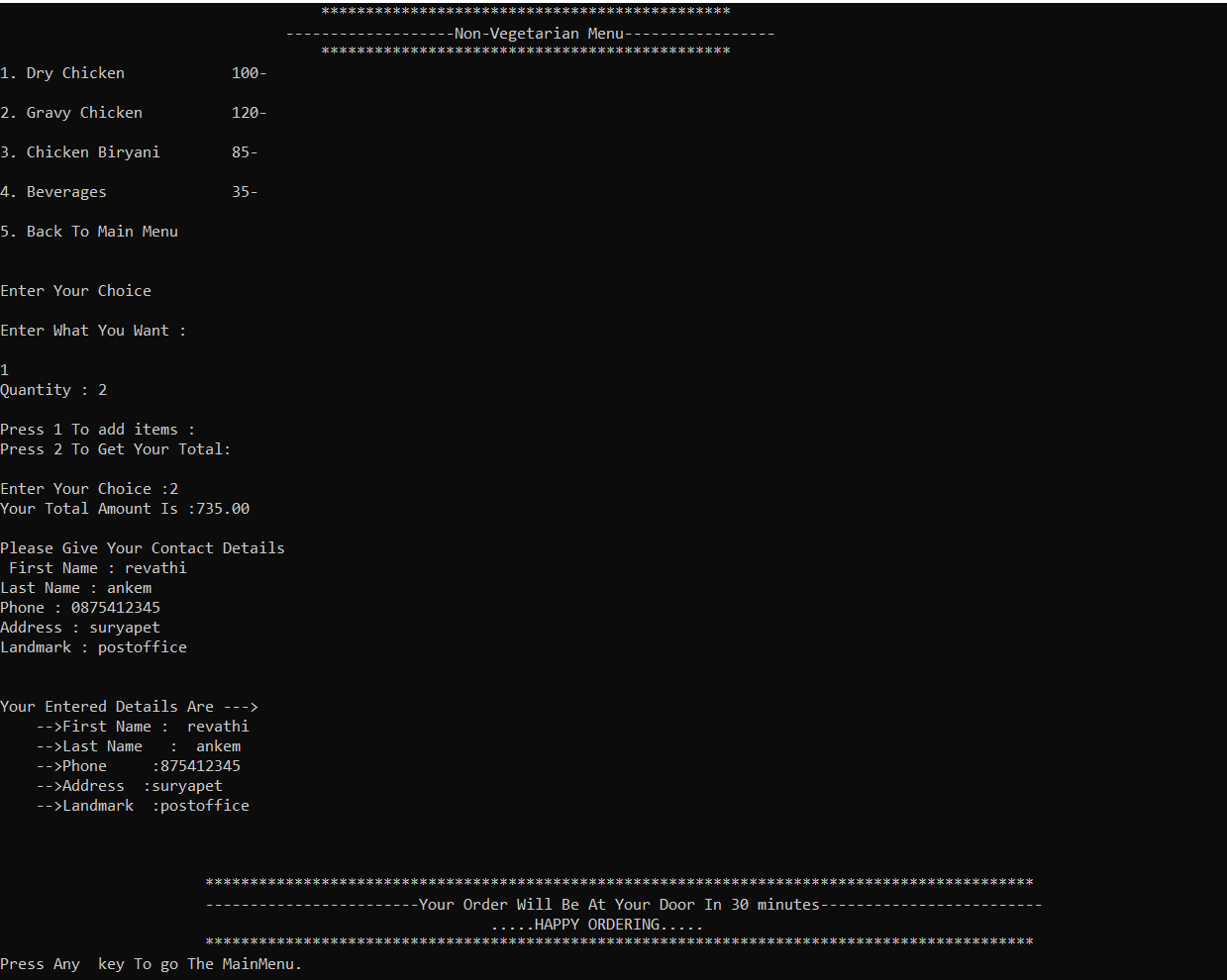
|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC01 | | | Use case ID :  **FC-10** |
| **Test case title** : Checks previous orders. | | |
| **Test case description** Allows admin to check the previous order details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin chooses history of orders option. | The system should display the history of order and details of customer. | Displays  The orders and details of customer. | |

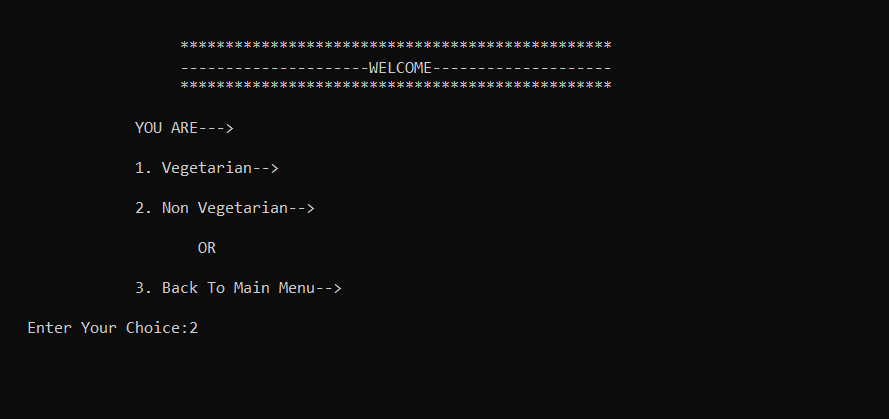
|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE TEMPLATE | | | |
| **Test case ID** : TC02 | | | Use case ID :  **FC-10** |
| **Test case title** : Checks previous orders. | | |
| **Test case description** Allows admin to check the previous order details. | | |
| **Test steps** | **Expected result** | **Actual result** | |
| Admin chooses history of orders option. | The system should display the history of order and details of customer. | Displays  “No orders” | |

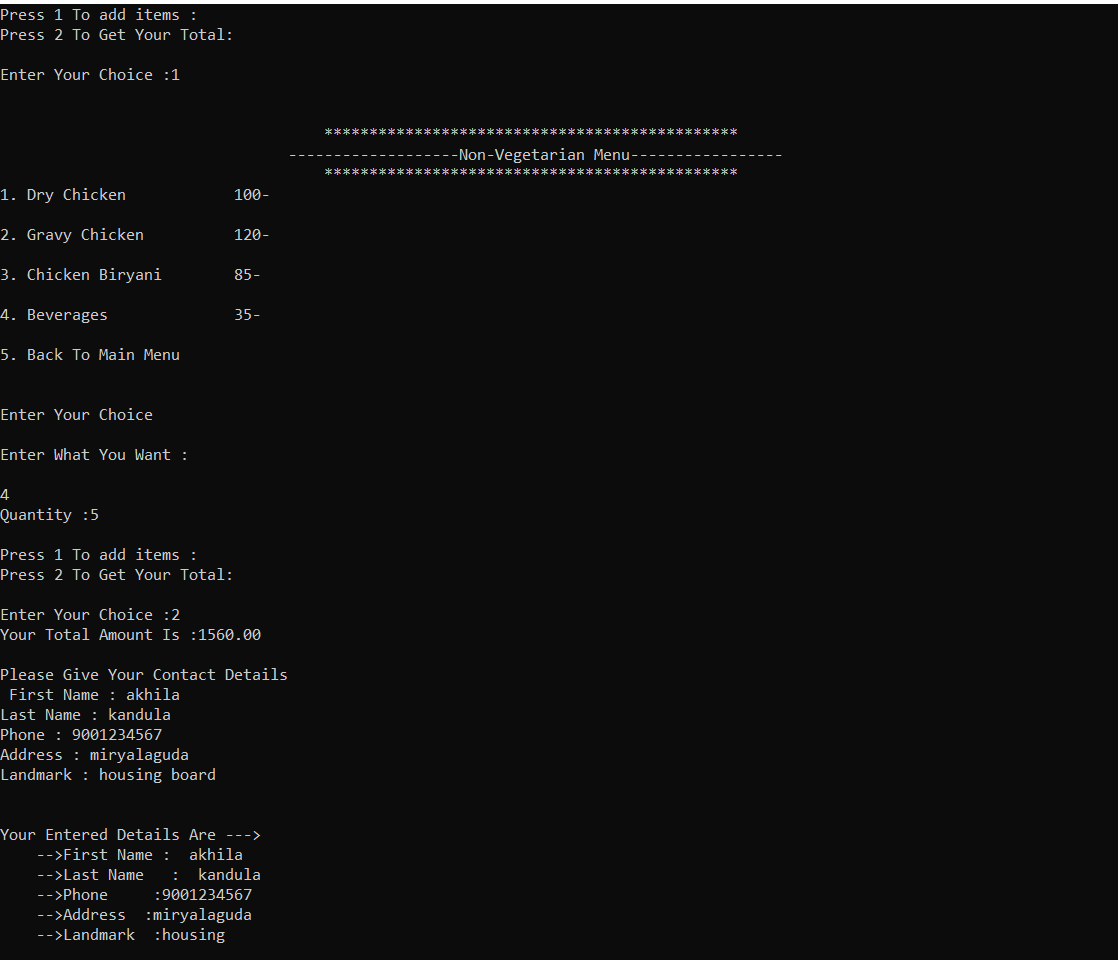
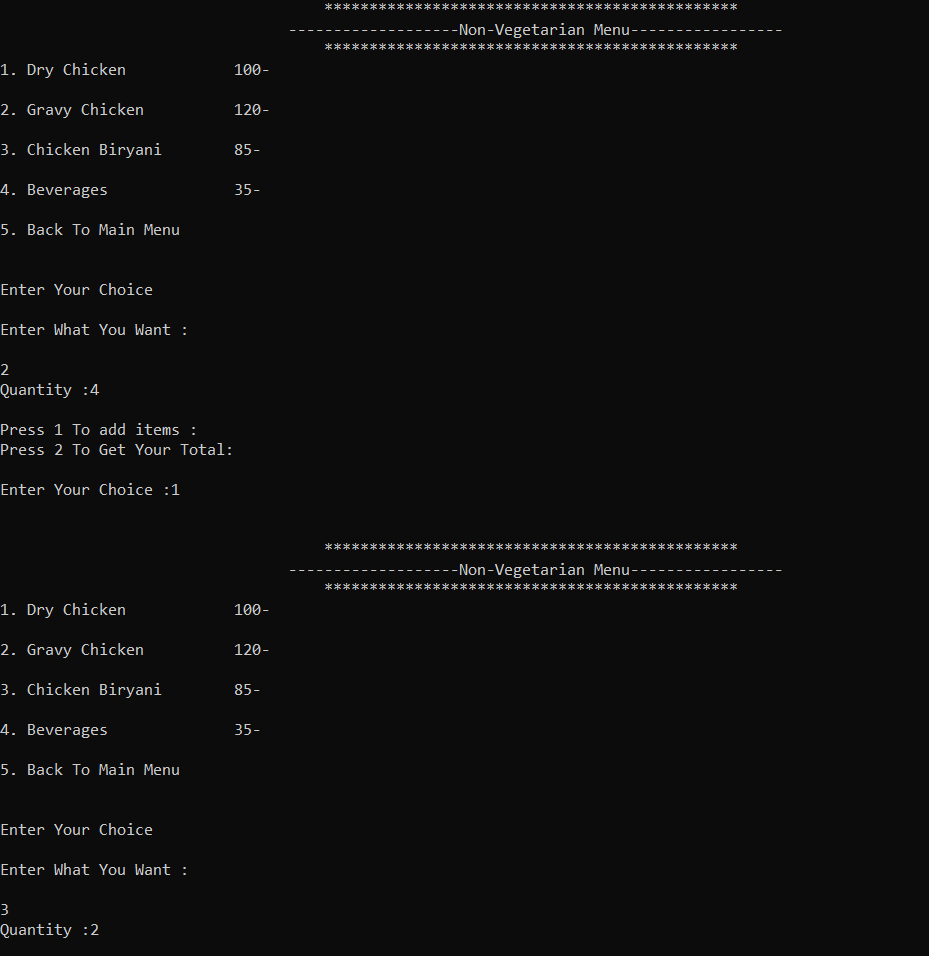


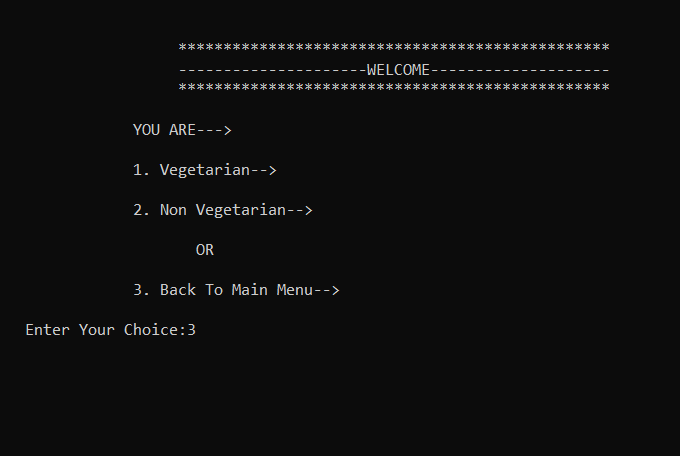
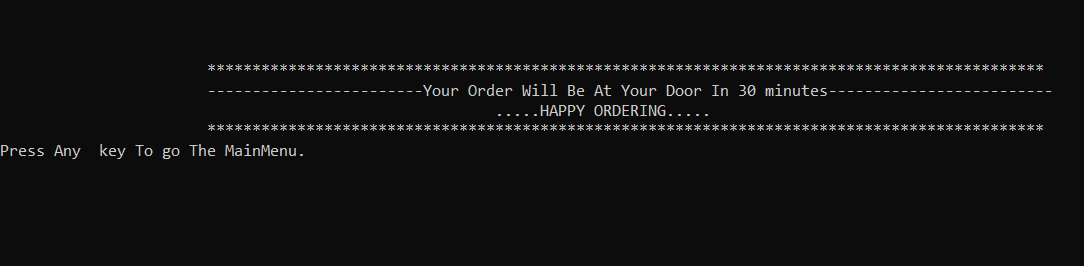


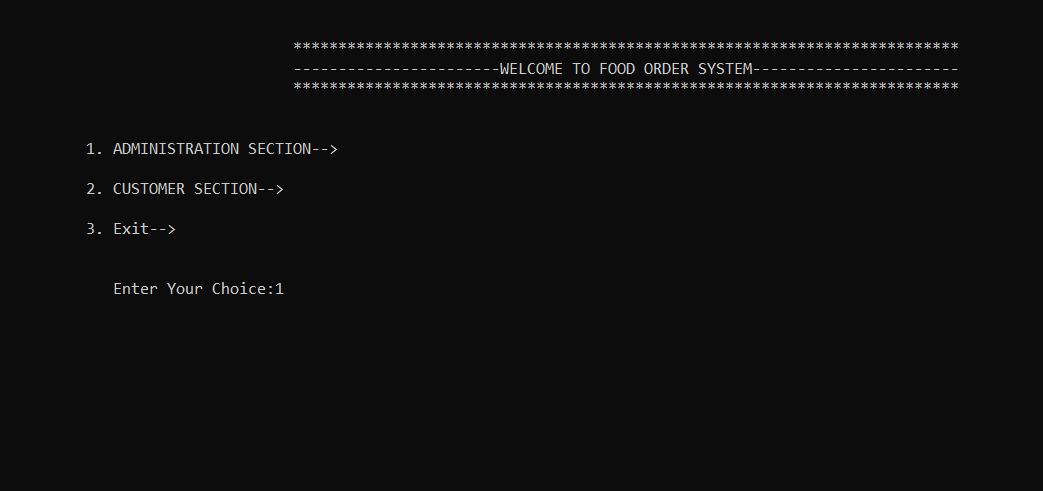


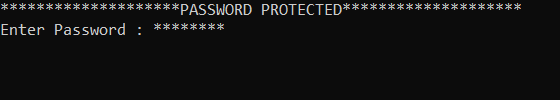


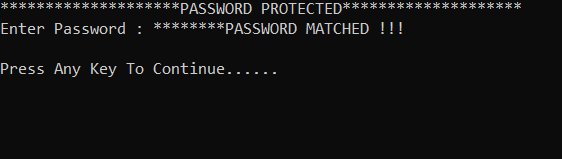


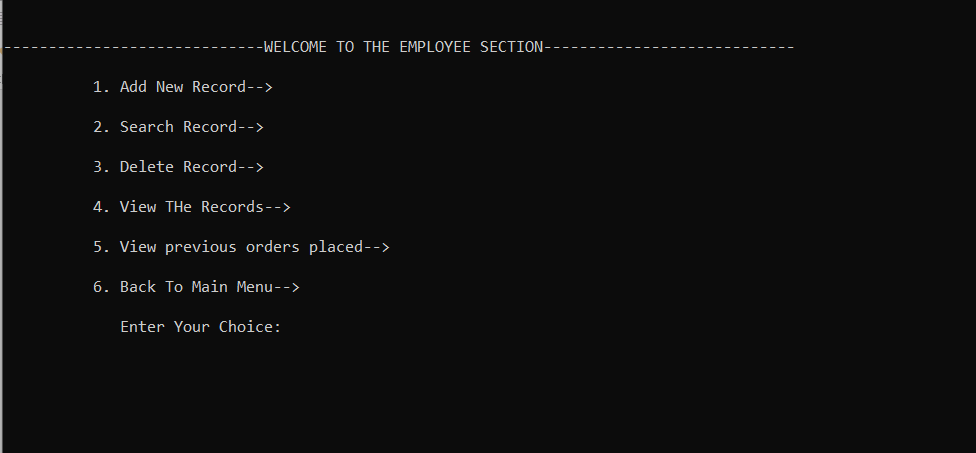


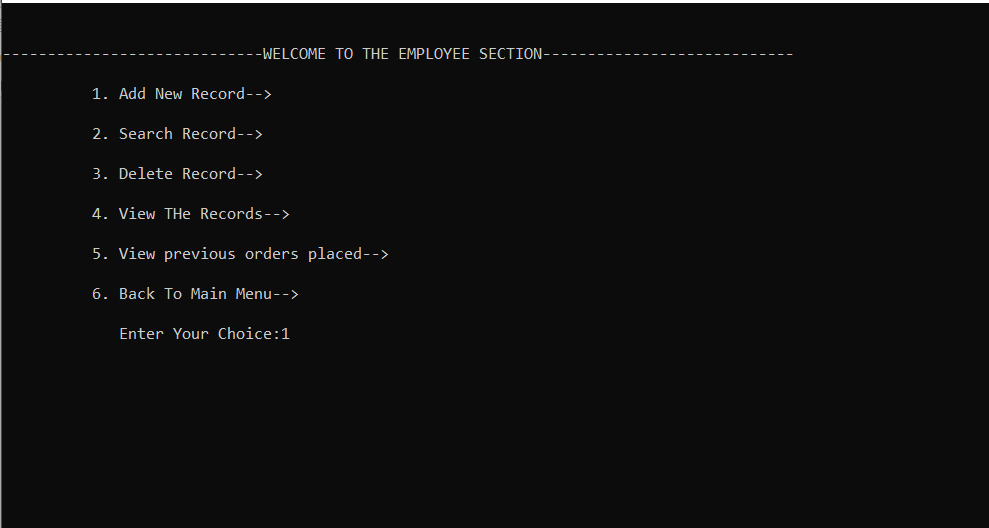


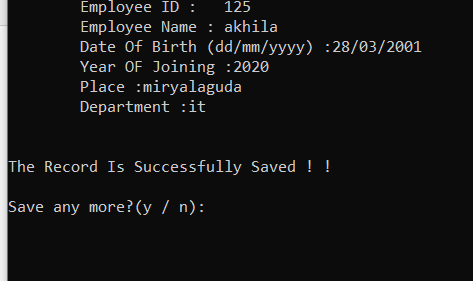


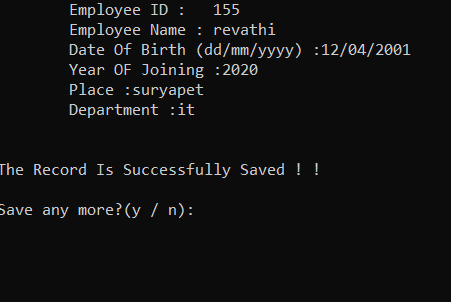


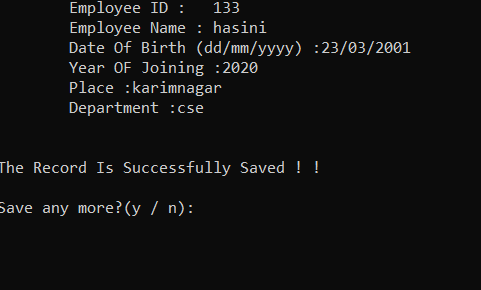




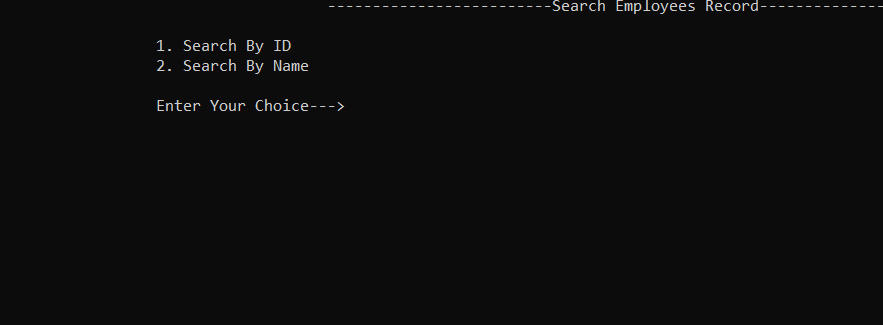


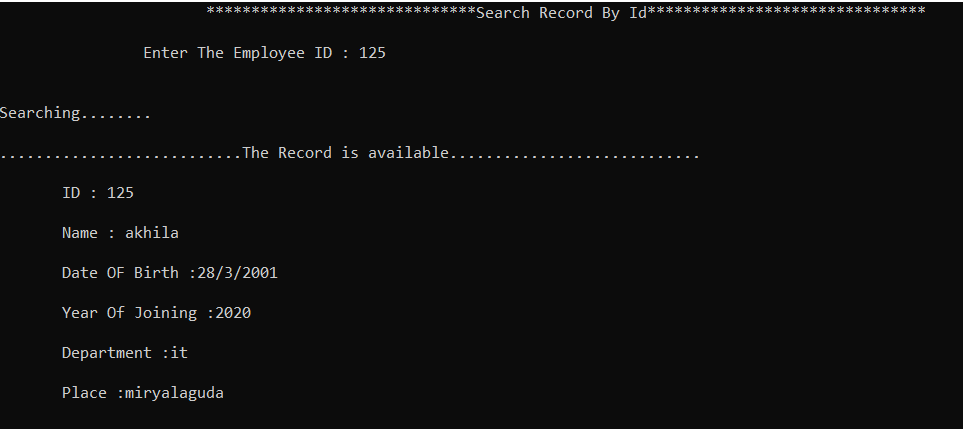


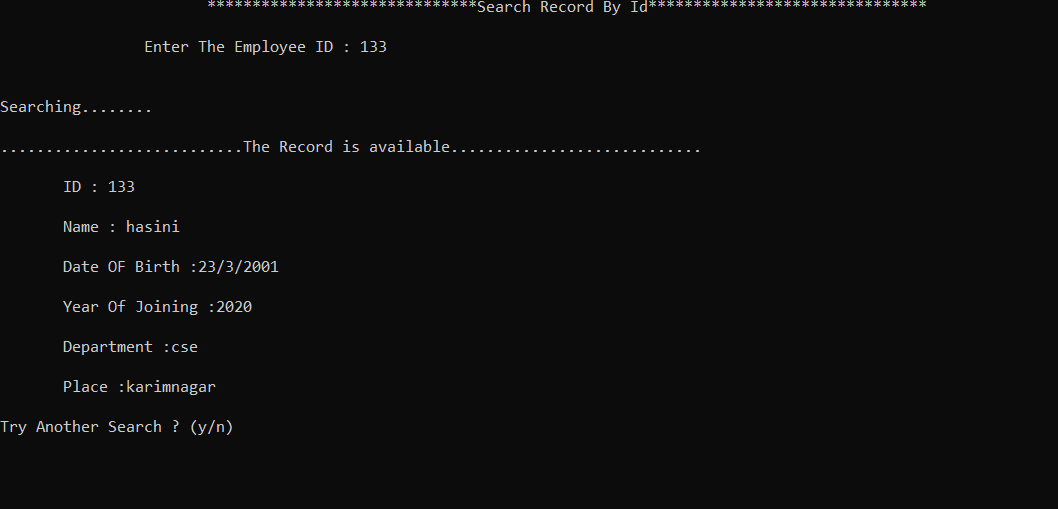


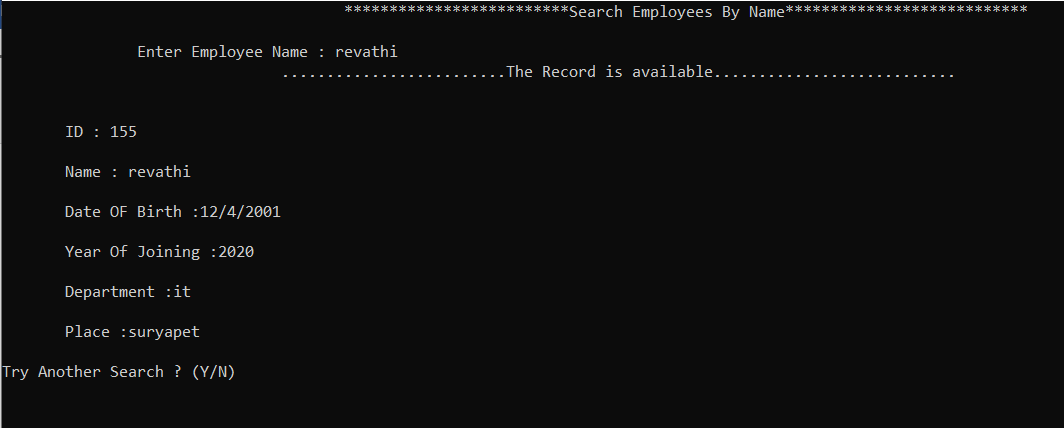


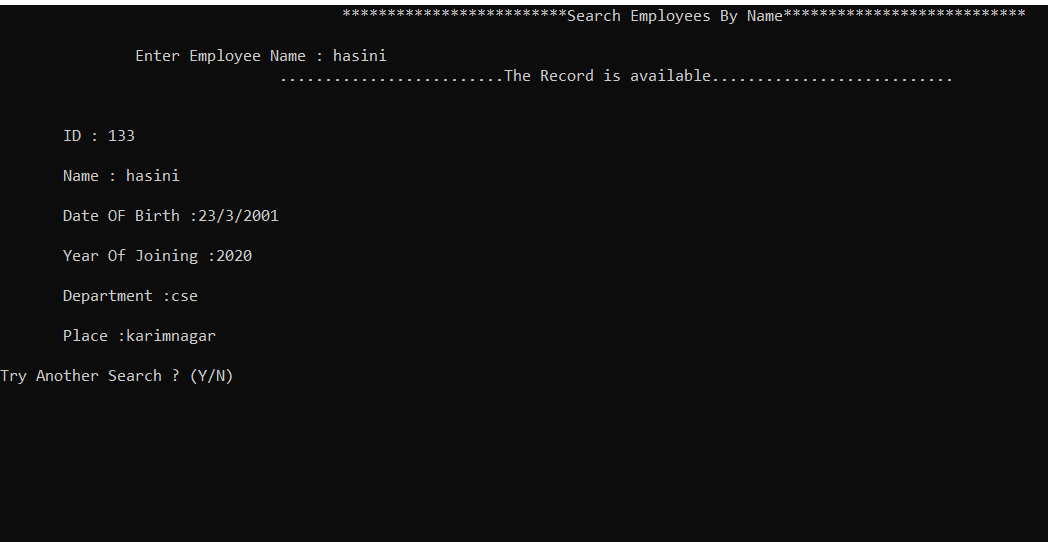




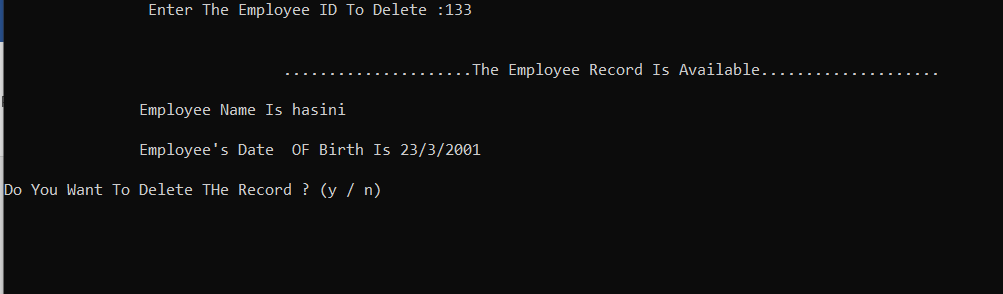


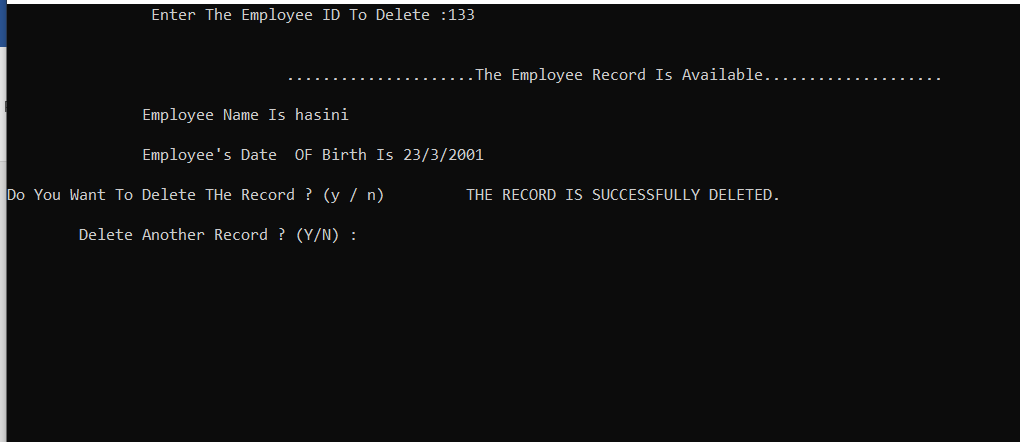


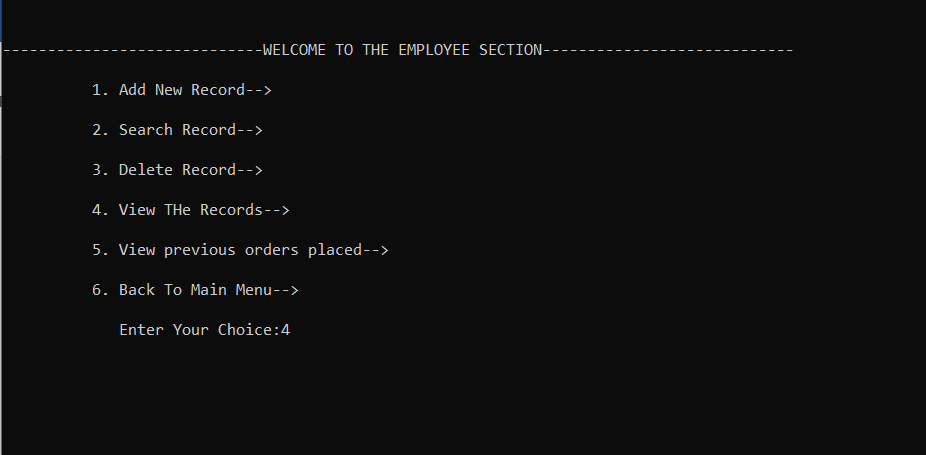


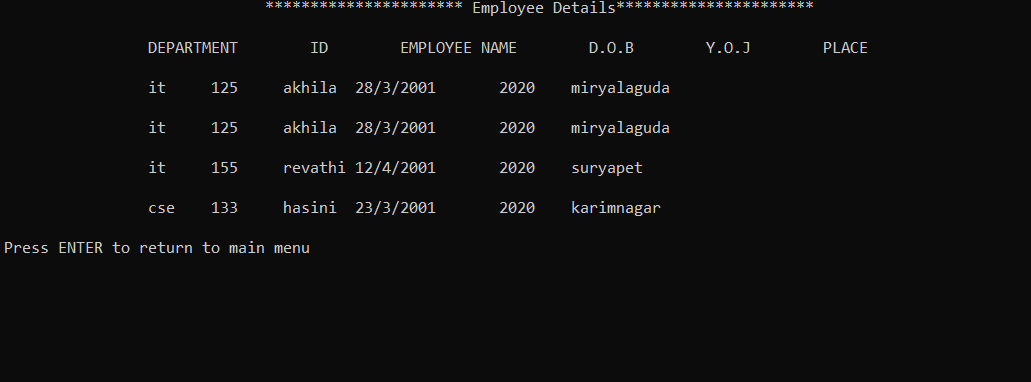


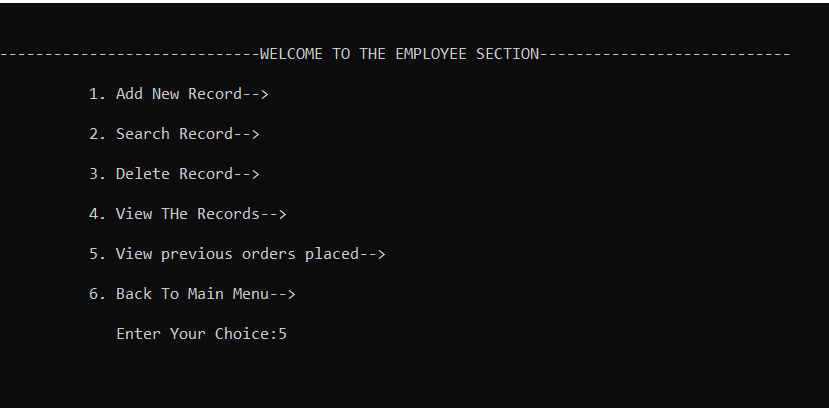














**What is the additional knowledge gained as a result of implementing this mini project apart from the syllabus covered in the course programming for problem solving?**

**CONCLUSION AND FUTURE WORK**

All the data will be saved in the database. So, the administer can view all the data on time. This system reduces manual works too.

The following section describes the word that will be implemented will future releases of the software.

Customize Orders: Allow customers to customize food orders

Enhance User Interface by adding more user interactive features.

Provide deals and promotional Offers details to home page.

Provide Recipes of the week / Day to Home page

**REFERENCES**

We referred google and our c textbook for the completion of project successfully.

The C Programming Language is written by Brian **W. Kernighan and Dennis M. Ritchie**. This book is considered to be the best C programming book because it is written by inventors of C language. The book explains the concept of C language in an easy manner and covers every concept in detail.

**www.geeksforgeeks.org**