Food Delivery Application

UCS2265 – Fundamentals and Practice of Software Development

A PROJECT REPORT

Submitted By

Kushaal Shyam Potta 3122235001071
Mahadev Ramesh Ramya 3122235001075
Murari Sreekumar 3122235001087

Department of Computer Science and Engineering

Sri Sivasubramaniya Nadar College of Engineering
(An Autonomous Institution, Affiliated to Anna University)
Kalavakkam – 603110

June 2024

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled "**Food Delivery Application**" is the bonafide work of "Kushaal Shyam Potta (3122235001071), Mahadev Ramesh Ramya (3122235001075) and Murari Sreekumar (3122235001087)" who carried out the project work in the UCS2265 – Fundamentals and Practice of Software Development during the academic year 2023-24.

Internal Examiner External Examiner

Date:

TABLE OF CONTENTS

PROBLEM STATEMENT:

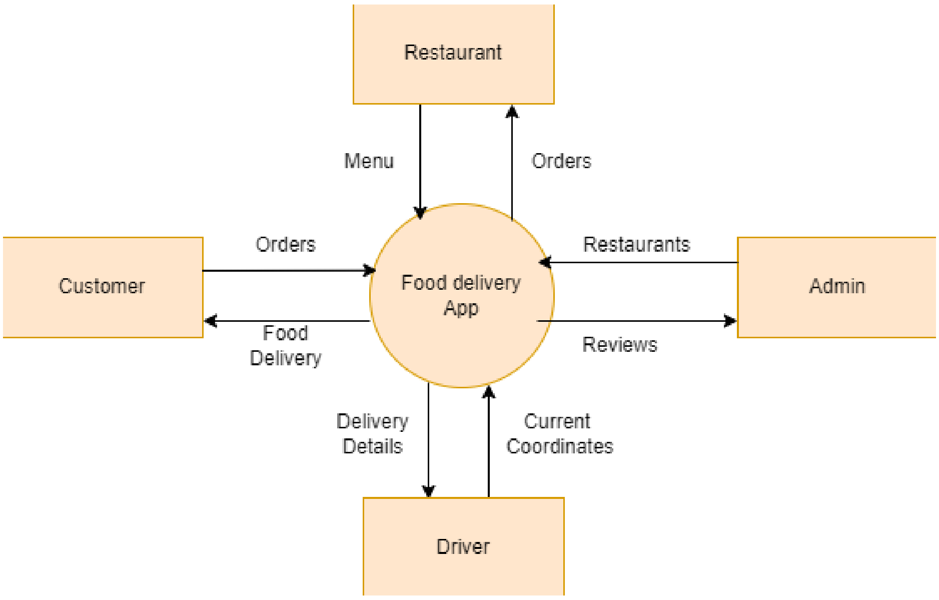
Develop an application for food delivery which facilitates ordering of food from restaurants of various cuisines from different locations. The user should be able to order the food through this application to any preferred delivery address.

EXTENDED EXPLORATION OF PROBLEM STATEMENT:

1. Restaurant Login: Creating a separate facility for the restaurants to login and change the menu according to their needs.
2. Calorie Tracker: Sends the calories of the corresponding items ordered by the user.
3. Membership Registration: The user is given an option of registering as a member of the app where the user is given loyalty points based on which he can get exclusive offers.
4. Delivery Login: Creating a separate facility for drivers to login and choose the area they want to work in.

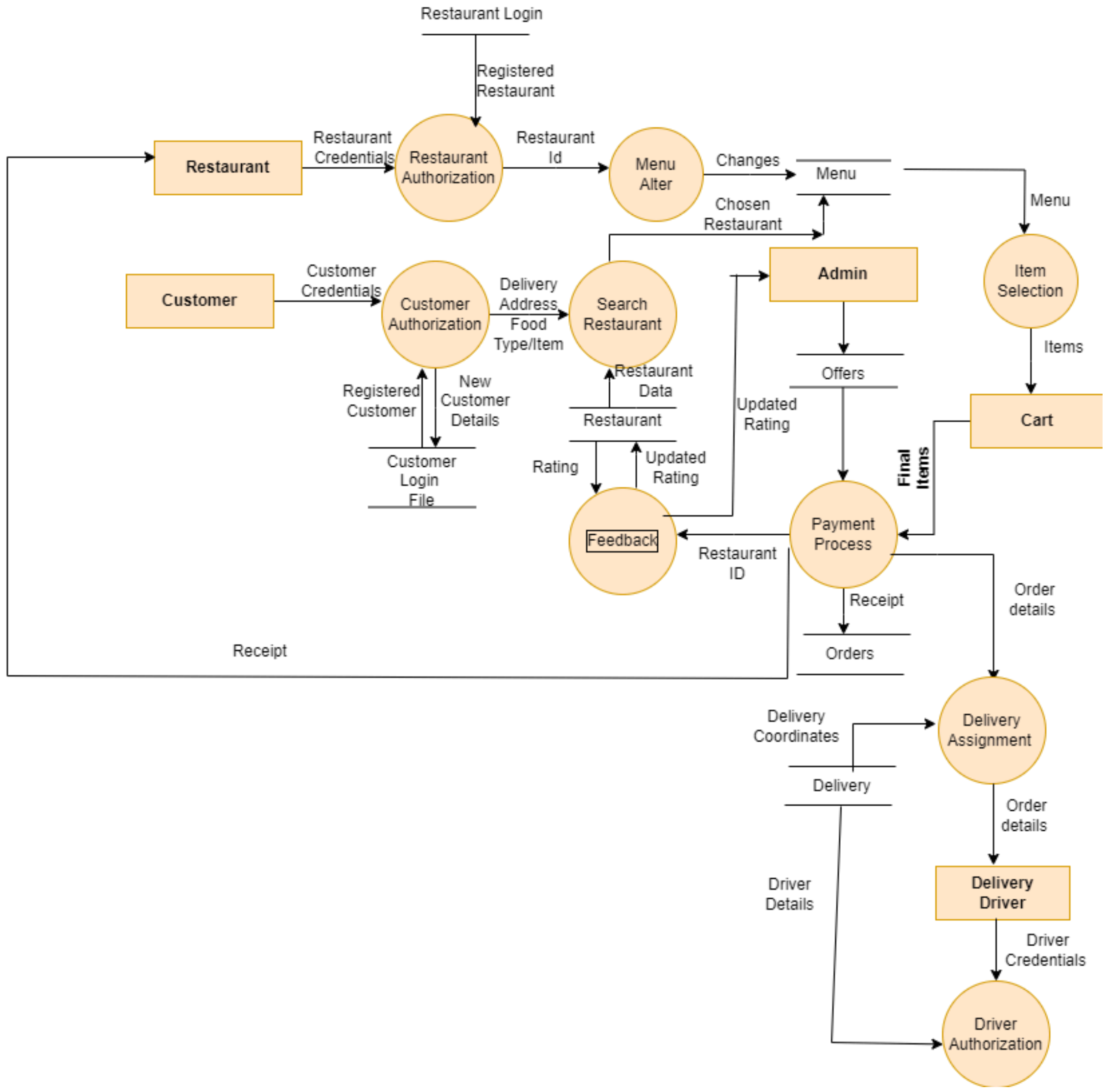
ANALYSIS USING DATA FLOW DIAGRAMS:

DATA FLOW DIAGRAM LEVEL 0:



- Customers interact with the app.
- App uses restaurant details to perform its functions.
- Admin is able to oversee the working of the app.
- Driver receives delivery details and their coordinates are sent for tracking.

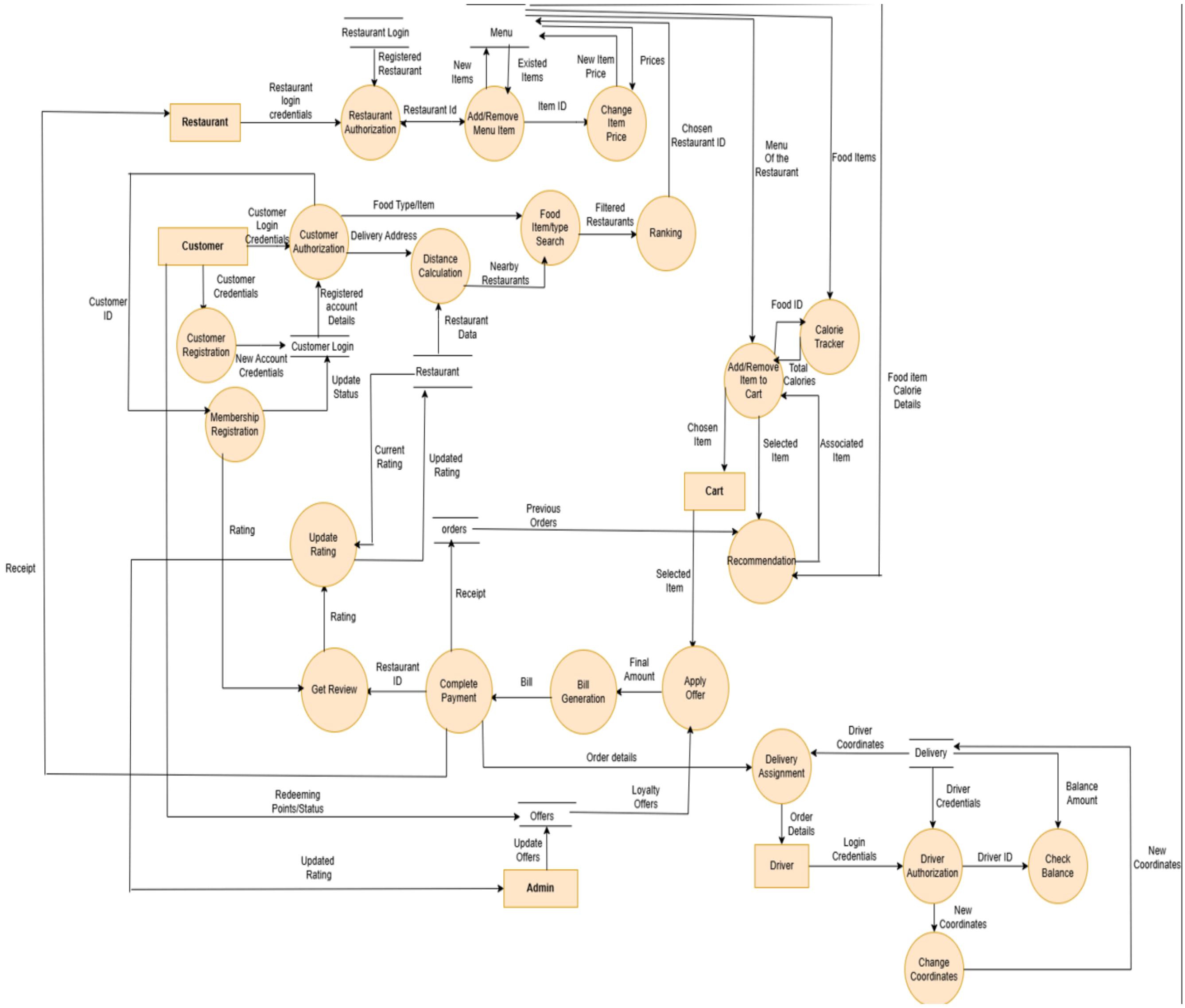
DATA FLOW DIAGRAM LEVEL 01:



- Customer Authorization: The customer sends his login credentials and it checks whether the login credentials matches with the one in the customer login database.
- Restaurant Authorization: Here the restaurant is allowed to login and as he sends restaurant credentials and it checks whether the login credentials matches with the one in the restaurant authorization database.
- Search Restaurant: Customers search which restaurant they want to choose.
- Menu Alter: Restaurants can alter their menu as per their requirements.
- Item Selection: User selects items from Menu and it is passed to the cart.
- Payment Process: This function handles the payment process through E-Wallet.
- Feedback: Feedback is taken from the user and it is used to update ratings of restaurants.
- Delivery Assignment: Assigns the order to the nearest delivery agent available.

- Driver Authorisation: The driver sends his login credentials and it checks whether the login credentials match with the one in the driver login database.

DATA FLOW DIAGRAM LEVEL 02:



- Membership Registration: Here the customer is given an option to register as a member so that he can receive exclusive offers based on his loyalty points.
- Add/Remove Item from the Menu: Here the restaurant is given the chance to alter the menu by adding /removing items of the restaurant.
- Change Item Price: Here the restaurant is given the option of changing the price of the items.
- Distance Calculation: Here the delivery address of the customer is sent as an input and restaurants within the 10km distance of the delivery address are sent to the customer using the shortest path algorithm.
- Food Item/type search: Here the food item /type of the user is sent as an input and the

restaurants are filtered accordingly.

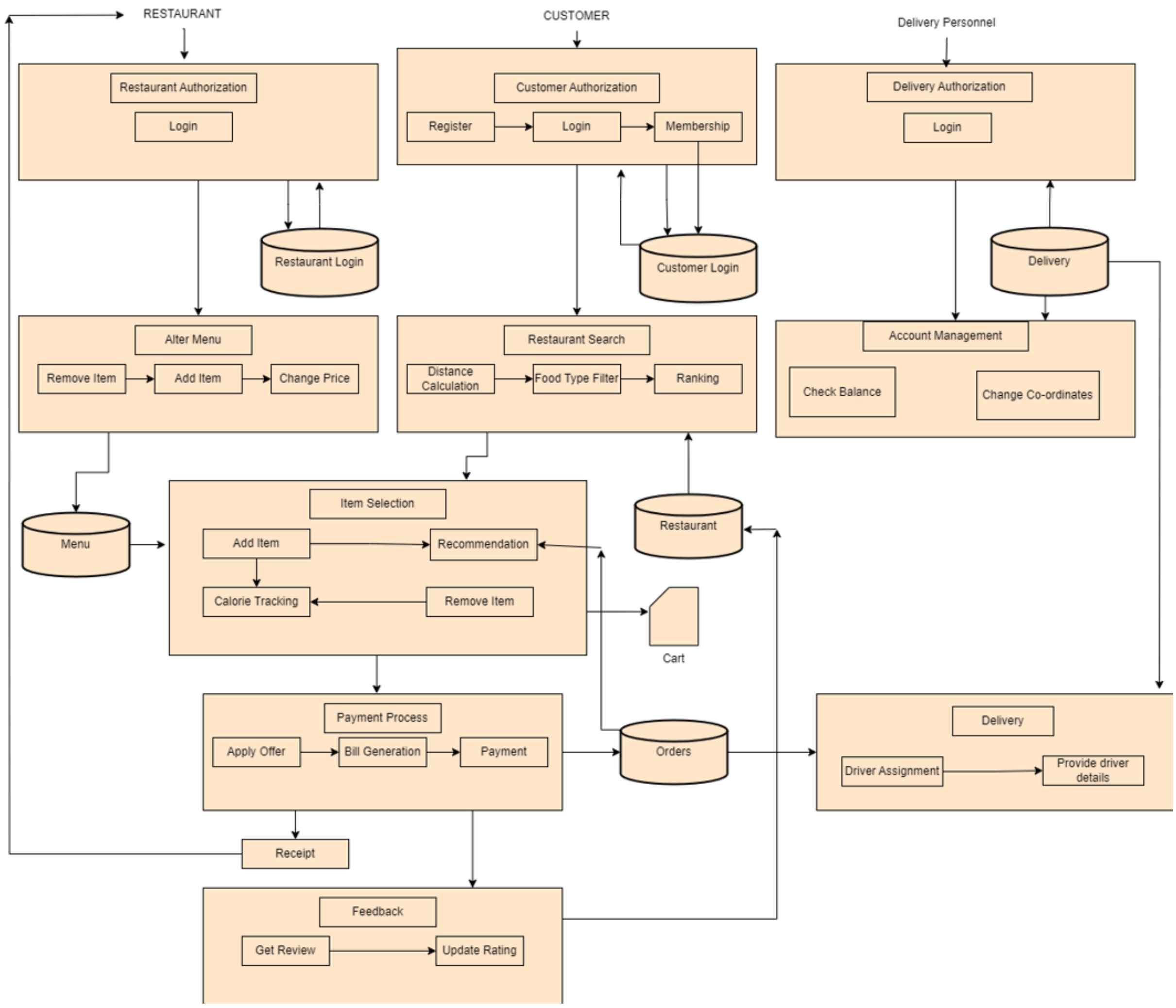
- Ranking: Here the restaurants are ranked based on parameters given by the user such as star ratings, special offers.

- Add/Remove item to the cart: Here the user is given the option to add / remove items to the cart based on the menu of the restaurant he receives. He is also given the option of viewing the cart.
- Recommendation: In recommendation based on the previous orders and selected items associated food items are returned using associative mining .
- Calorie Tracker: Here the food id is sent to the calorie tracker where the calories of the corresponding items are sent back to the user while ordering items.
- Apply Offers: In apply offers based on the items given by the user general offers are applied to reduce the total amount and in case the user was a member based on the loyalty points points +are also applied to reduce the amount and the reduced amount is sent to the bill generation module.
- Bill Generation: Here the bill is generated and sent to the complete payment module.
- Complete Payment: user completes the payment by deducting the amount from the e-wallet.

- Get Review: In Get Review based on the restaurant id given by the user the user is asked to give a 5-star rating based on the food he received from the restaurant.
- Update Rating: In Update Rating the current rating of the restaurant is sent from the restaurant database and the updated rating is sent back to the restaurant database.
- Check Balance: displays the bank balance to the driver .
- Change coordinates: the driver is given an option to change his position by changing the coordinates.

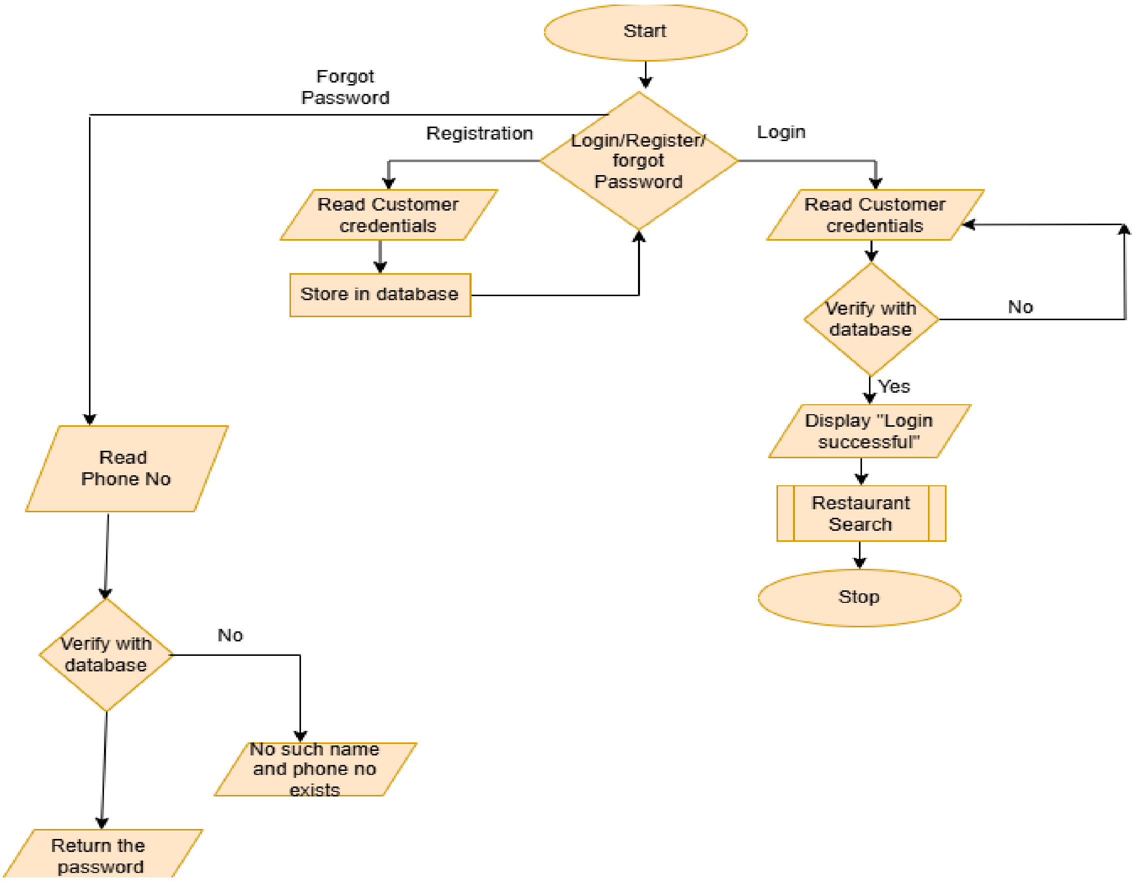
DETAILED DESIGN:

ARCHITECTURE DIAGRAM:



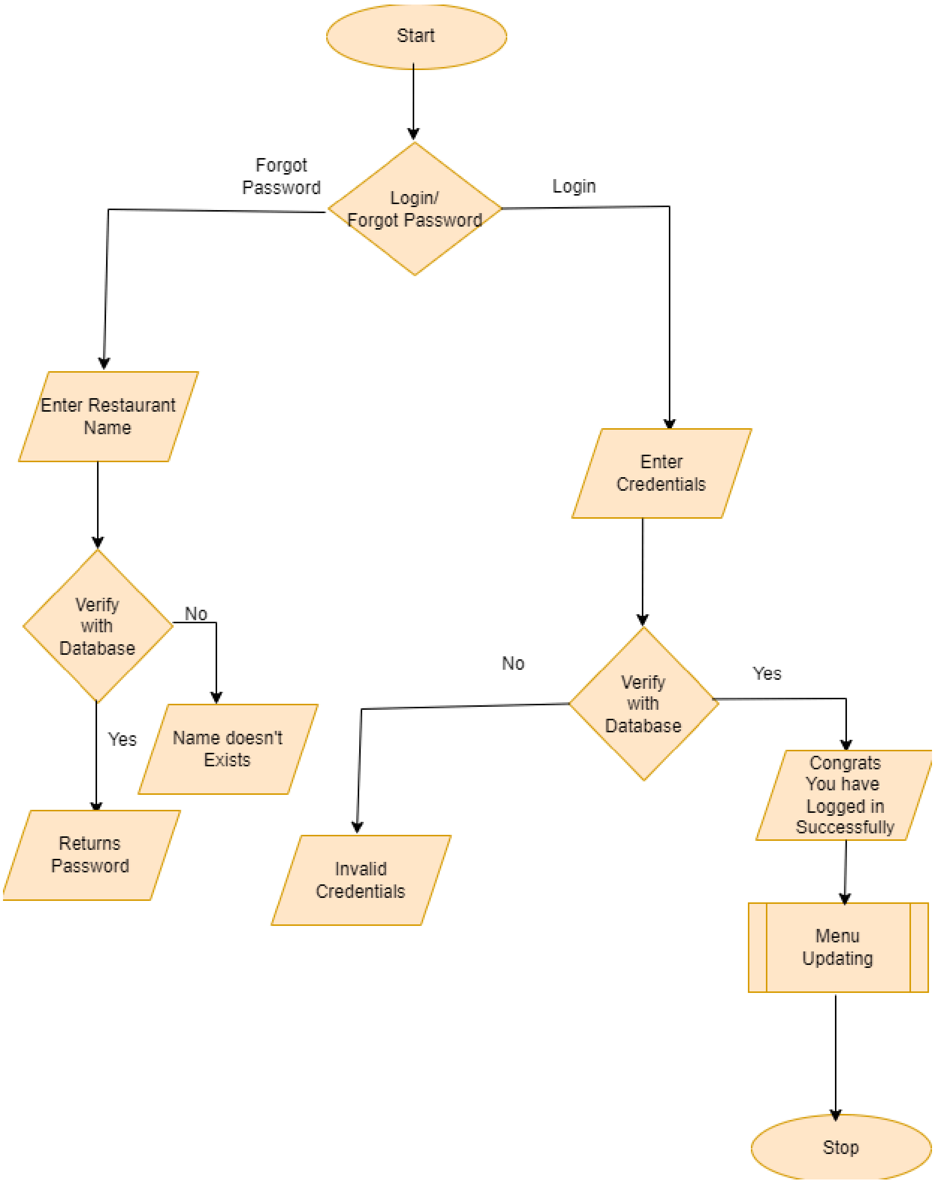
MODULES USED:

CUSTOMER AUTHORIZATION/REGISTRATION:



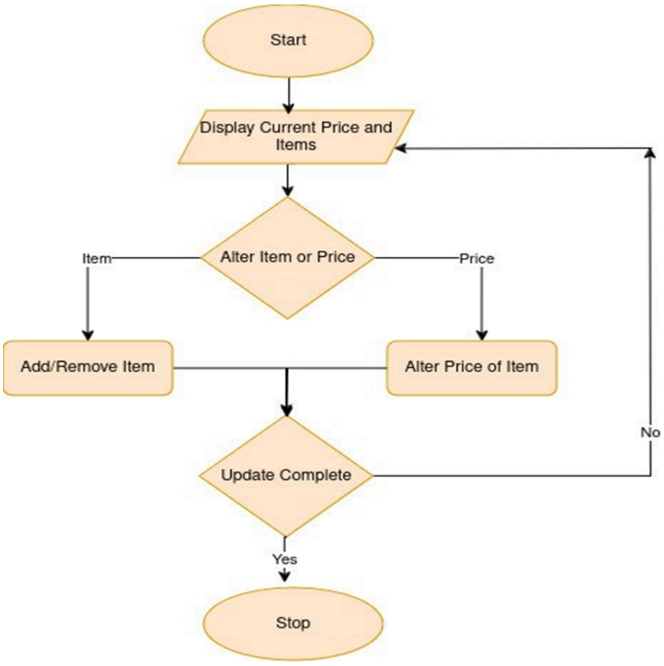
- Information of all registered customer accounts are stored in ‘Customer Login’ database.
- The customer is prompted to choose between login or registration, if a first-time user.
- In registering a new account, the customer will be asked to create a username and password which would be stored in the ‘Customer Login’ database, after which they would be redirected to the login page.
- In login, the customer must enter a valid username and password, which would be validated with the database. If the details are correct, then he would be forwarded to the next page. If incorrect, then they would be asked to re-enter the credentials.
- After a successful login, non-members would be given the details of exclusive membership and the opportunity to purchase it.

RESTAURANT LOGIN:



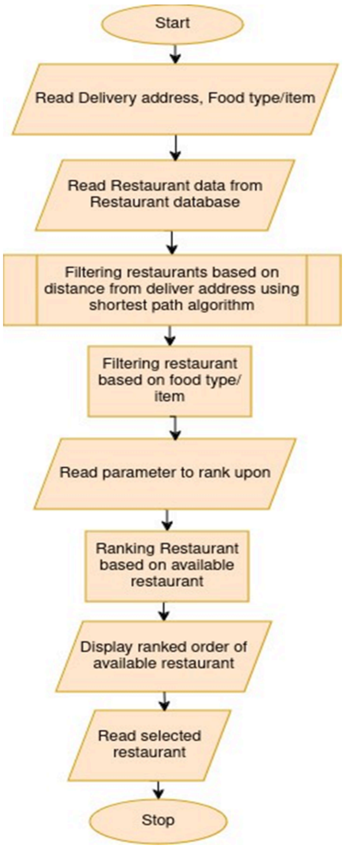
- Information on restaurant user credentials are stored in ‘Restaurant Login’ database.
- The user would be required to enter his credentials, which would be validated with the database.
- If the entered details are correct, they would be forwarded to the next page, else they would be asked to re-enter the details.

MENU UPDATE:



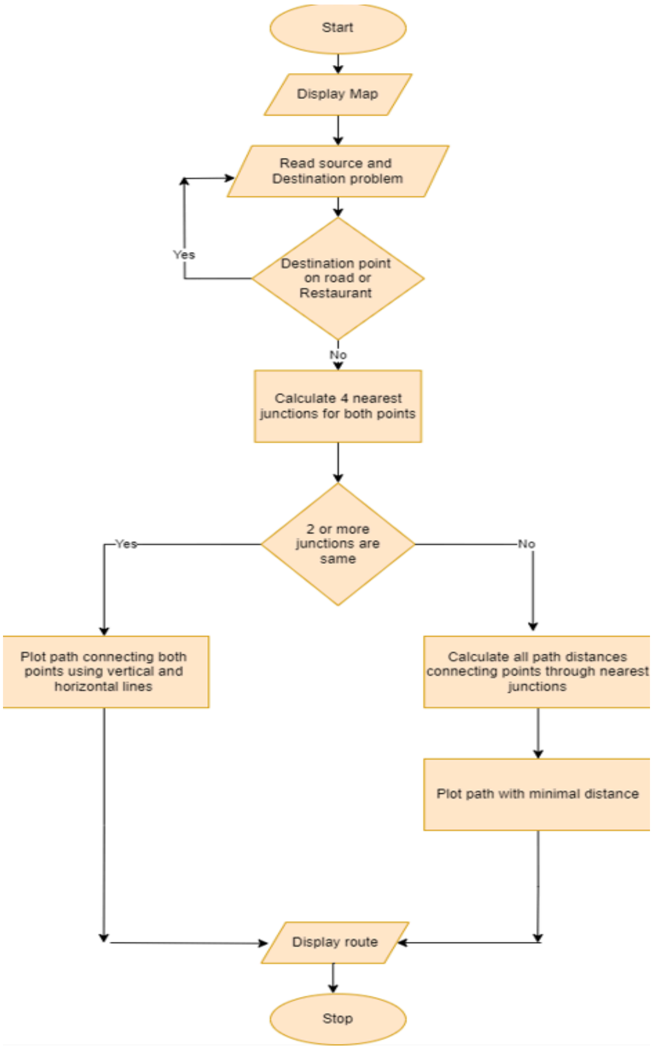
- All menu information is stored in the ‘Menu’ database.
- Here, the restaurants would be able to add/remove any item in the existing menu.
- They could also change the price of any item.

RESTAURANT SELECTION:


Start
Read Delivery address, Food type/item
Read Restaurant data from Restaurant database
Filtering restaurants based on distance from deliver address using shortest path algorithm
Filtering restaurant based on food type/ item
Read parameter to rank upon
Ranking Restaurant based on available restaurant
Display ranked order of available restaurant
Read selected restaurant
Stop

- The customer is prompted to enter their delivery address and their desired food type.
- A primary filter of restaurants closer than 30-blocks from the delivery address is done first, followed by a secondary filter satisfying their desired food type.
- An item-based search is also possible, where the secondary search would filter restaurants with the available specific item.
- The satisfying restaurants would then be ranked by either distance, rating or offers based on the customer's choice.

SHORTEST PATH:


Start
Display Map
Read source and Destination problem
Yes
Destination point on road or Restaurant
No
Calculate 4 nearest junctions for both points
Yes
2 or more junctions are same
No
Plot path connecting both points using vertical and horizontal lines
Calculate all path distances connecting points through nearest junctions
Plot path with minimal distance
Display route
Stop

- The source and destination coordinates are read.
- The four nearest junction nodes for each point is calculated and the distance it takes to travel there is also counted.
- The source junction nodes and destination junction nodes are paired up and the distances between each of them is calculated.
- The minimal sum of the distance to travel to junction node and from source junction node to destination junction node is returned.

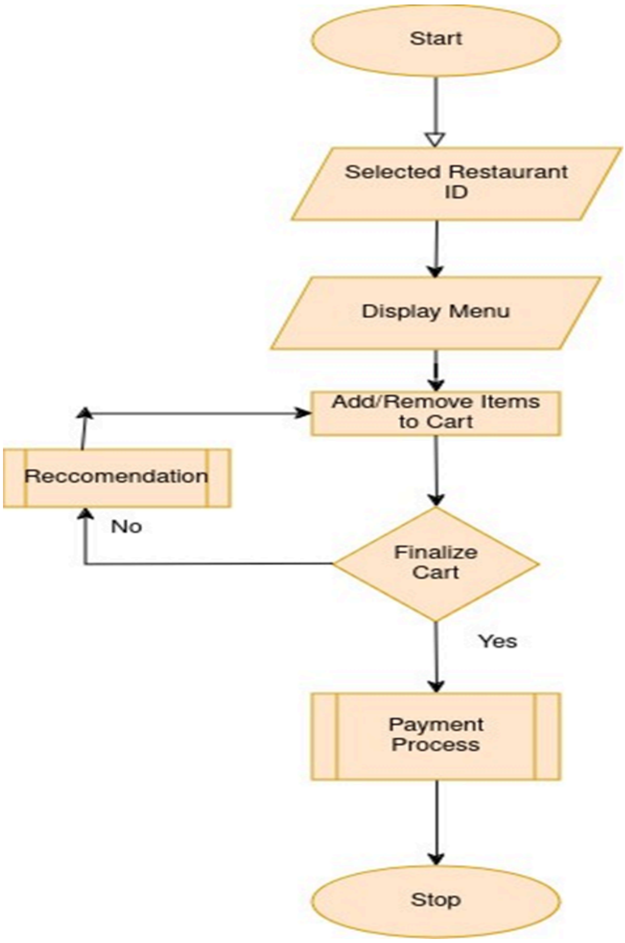
Comparative Study of Shortest Path Module:

The algorithm's working is based on the Floyd-Warshall algorithm. Due to the simplicity of our map, we have altered the algorithm to work only for positive weights and assume every path between any pairs of nodes is bi-directional.

Supposing in our map, we wish to find the shortest distance between points x and y. We first check if the two points are in neighboring districts, if so we calculate the shortest path with horizontal and vertical difference between them.

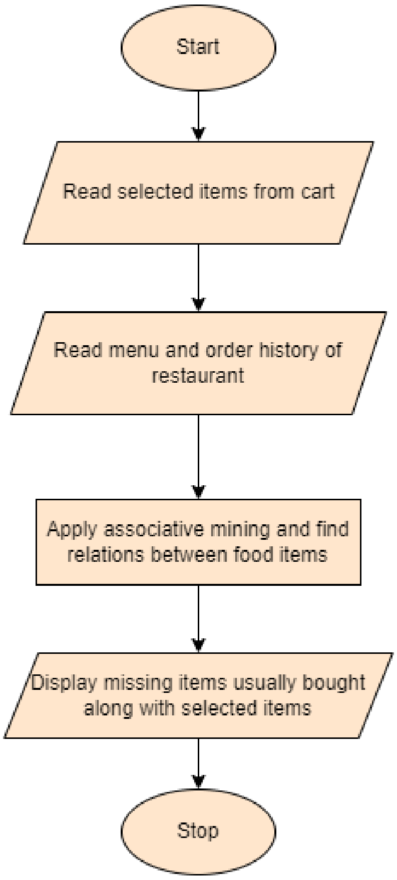
If they aren't in neighboring districts, we first find the 4 nearest junctions surrounding them, x1,x2,x3,x4 and y1,y2,y3,y4 for points x and y respectively. We then find the shortest path between each node pair such as (x1,y1),(x1,y2),(x2,y1),(x2,y2),etc.

ITEM SELECTION:



- The selected restaurant's menu is retrieved from the 'Menu' database by using the restaurant ID as key.
- The user can add/remove items from the temporary 'cart' file created.
- Based on the selected items, the current available items in the menu and the previous order histories taken from the 'Orders' database, associated items are recommended.
- The finalized items in 'cart' are forwarded to the payment process.

FOOD RECOMMENDATION:


Start
Read selected items from cart
Read menu and order history of restaurant
Apply associative mining and find relations between food items
Display missing items usually bought along with selected items
Stop

- The current items in the menu are read, and stored as an array of item_asoc structures.
- Each line of previous orders database is traversed, and cross checked to see if there are any items from current cart items in the line.
- If so, a default 10 associativity points are added to the other items present along with it. If it was ordered by the same customer as current, then an additional 5 points are added.
- The above process is repeated for every line of the database.
- The top ranked items based on associativity points which aren't in the cart are recommended to the user.

Comparative Study of Food Recommendation:

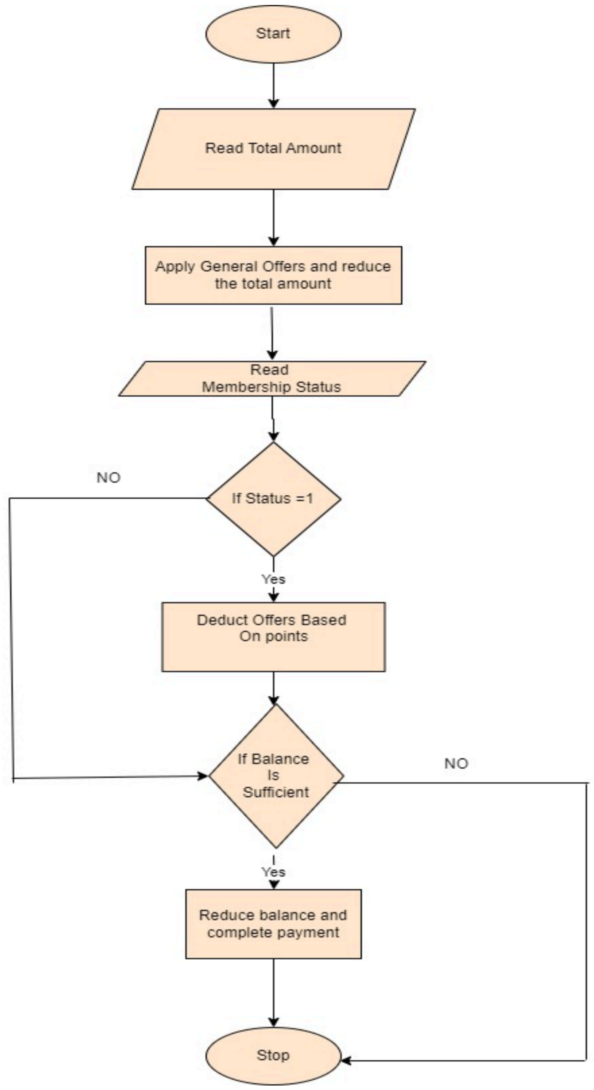
We have based our recommendations model from Apriori algorithm. However, due to the lack of sufficient database size, we have altered the algorithm to fit our needs.

We have removed a support count, and instead utilized a ranking method. Unlike in Apriori, there are no sets made beforehand, instead we see the frequency of associativity of an item with all the selected items in the cart.

An advantage of the algorithm is that we are able to give higher weightage to items previously ordered by the customer when recommending.

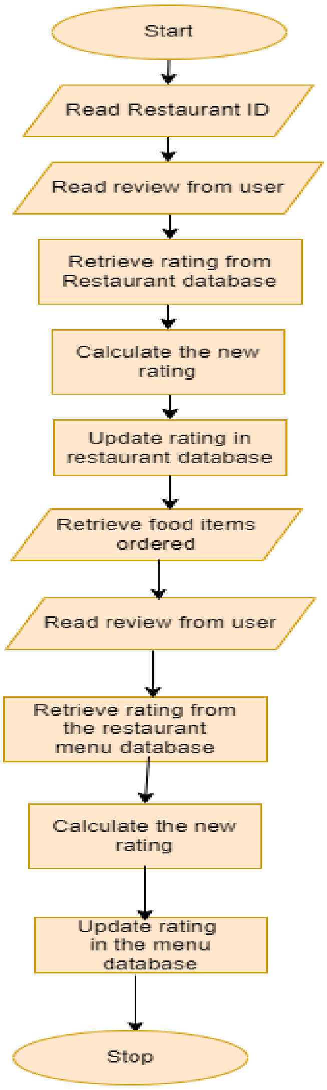
The code has high complexity requiring a lot of space and time, thus it is not suitable for large databases.

PAYMENT:


Start
Read Total Amount
Apply General Offers and reduce the total amount
Read Membership Status
If Status =1
NO
Yes
Deduct Offers Based On points
If Balance Is Sufficient
NO
Yes
Reduce balance and complete payment
Stop

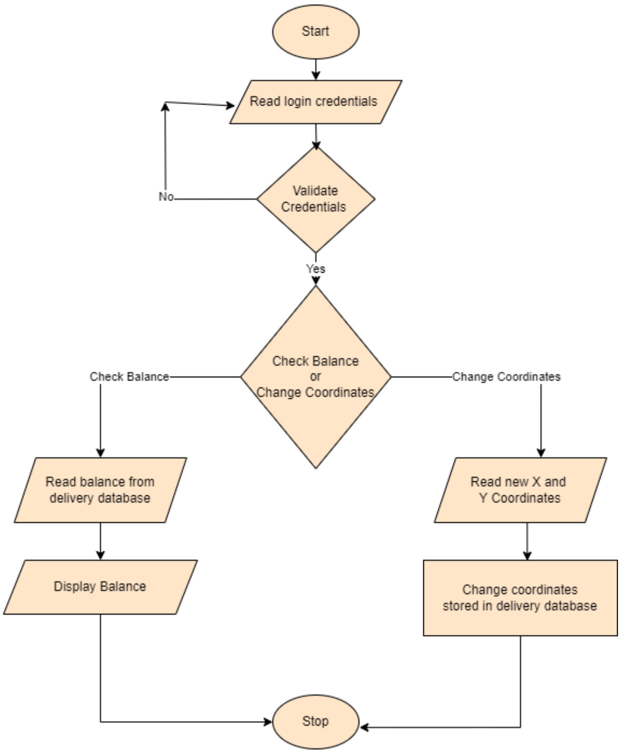
- First general offers are applied on the total amount based on the restaurant.
- Then based on his membership status he is given a chance to reduce total cost based on points.
- Then the balance is checked, If it is sufficient then its amount is reduced from the ewallet of the user, else print that there is not enough balance.

FEEDBACK:


Start
Read Restaurant ID
Read review from user
Retrieve rating from Restaurant database
Calculate the new rating
Update rating in restaurant database
Retrieve food items ordered
Read review from user
Retrieve rating from the restaurant menu database
Calculate the new rating
Update rating in the menu database
Stop

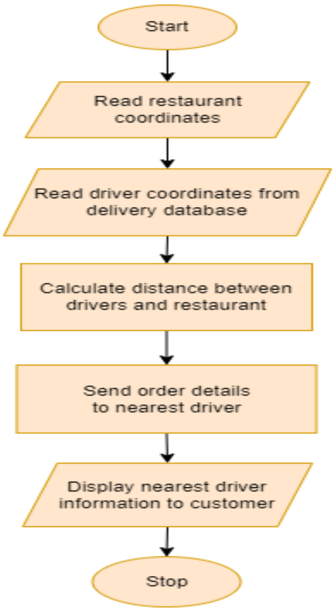
- The restaurant ratings are retrieved from the ‘Restaurant’ database.
- The customer is prompted to enter their feedback for their order and experience.
- The restaurant’s rating is updated with the new feedback in the database.
- The food items are retrieved from the food item database
- The user is asked to give the ratings of the food items
- The food items rating is updated in the database.

DELIVERY AUTHORIZATION:


Start
Read login credentials
Validate Credentials
No
Yes
Check Balance or Change Coordinates
Check Balance
Change Coordinates
Read balance from delivery database
Read new X and Y Coordinates
Display Balance
Change coordinates stored in delivery database
Stop

- The driver is first required to enter his account credentials, which are cross validated with the data stored in the delivery database.
- If the credentials are not matching, then he is requested to try again. If they are matching, then he is forwarded to the next page.
- Here, the driver has the option to either check his balance or change his station coordinates.
- If he wishes to check his balance, then the balance is retrieved from the delivery database.
- If he wishes to change his coordinates, then the new coordinates are read and stored in the database.

DELIVERY ASSIGNMENT:


Start
Read restaurant coordinates
Read driver coordinates from delivery database
Calculate distance between drivers and restaurant
Send order details to nearest driver
Display nearest driver information to customer
Stop

- The restaurant coordinates are received from the payment process.
- All the driver coordinates are read from the delivery database and the distance to the restaurant is calculated.
- The driver nearest to the restaurant is assigned to deliver the order,
- The order details are sent to the driver and the driver details are displayed to the customer.

IMPLEMENTATION:

Data organization:

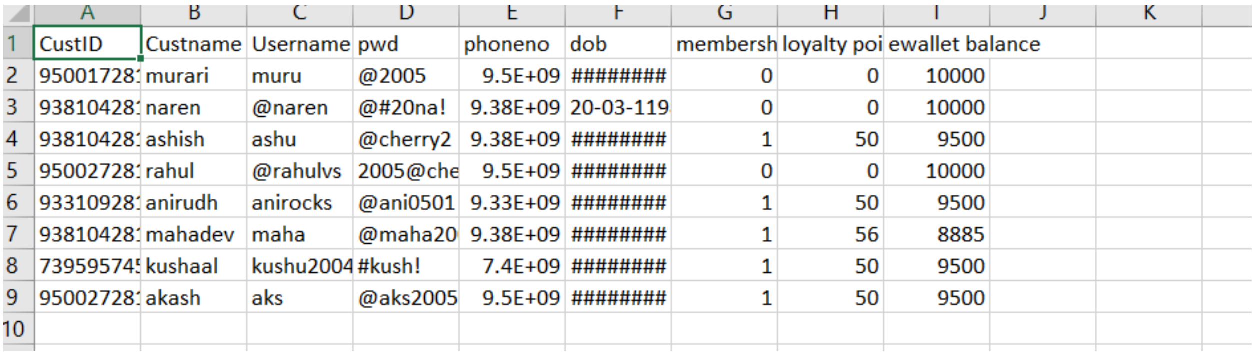
Here for the food delivery system module the required data are:

1) Customer credentials
2) Restaurant credentials
3) Menu for each restaurant
4) Previous order histories
5) Delivery driver details

So here the data is organized in the form of csv files as csv files are faster and better to handle.

Customer Credentials:

Customer credentials are stored in the form of customerlogs.csv which has the following columns:



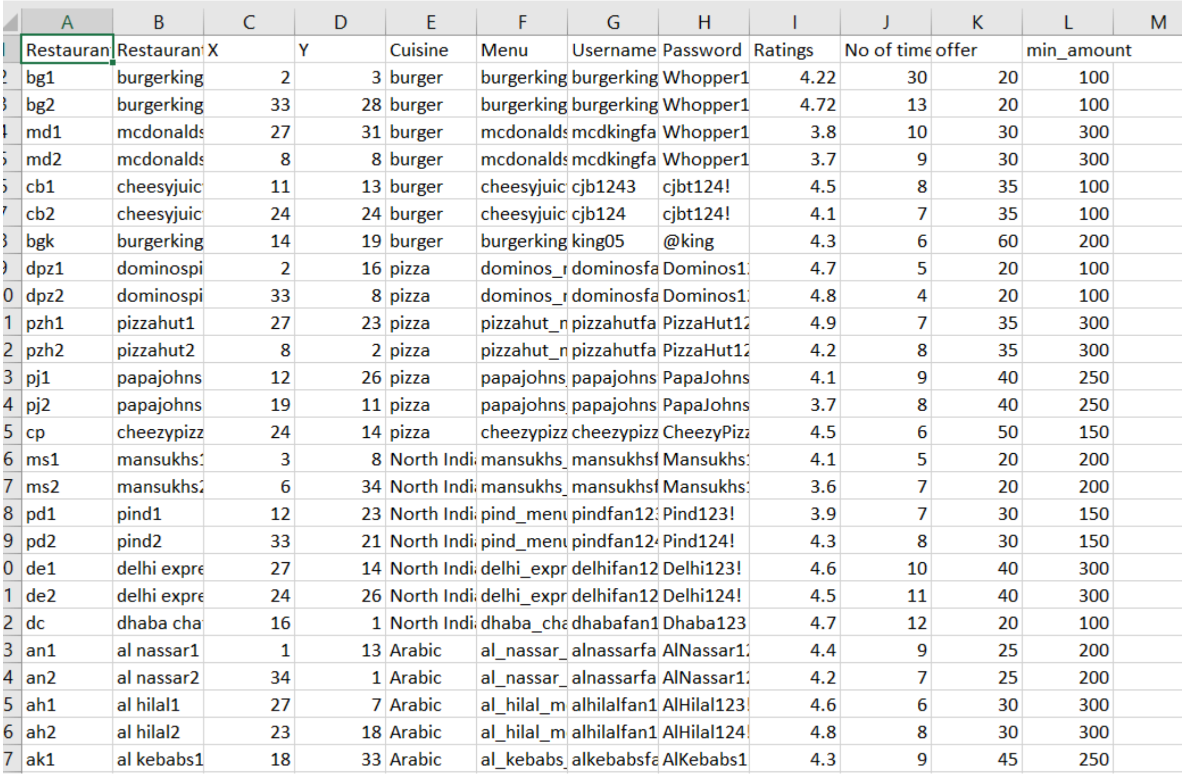
	A	B	C	D	E	F	G	H	I	J	K
1	CustID	Custname	Username	pwd	phoneno	dob	membersh	loyalty poi	ewallet balance		
2	95001728	murari	muru	@2005	9.5E+09	########	0	0	10000		
3	93810428	naren	@naren	@#20na!	9.38E+09	20-03-119	0	0	10000		
4	93810428	ashish	ashu	@cherry2	9.38E+09	########	1	50	9500		
5	95002728	rahul	@rahulvs	2005@che	9.5E+09	########	0	0	10000		
6	93310928	anirudh	anirocks	@ani0501	9.33E+09	########	1	50	9500		
7	93810428	mahadev	maha	@maha20	9.38E+09	########	1	56	8885		
8	73959574	kushaal	kushu2004	#kush!	7.4E+09	########	1	50	9500		
9	95002728	akash	aks	@aks2005	9.5E+09	########	1	50	9500		
10											

1) CustId: Contains the customer id. It is of string type. Created using the phone number and the name of the person.
2) Custname: Contains the name of the customer. It is of string type.
3) Username: It is of String type.
4) Password : It is of string type.
5) Dob: it is of string type (DD-MM-YYYY).
6) Membership status: it is of integer type. 1 means the customer is a member of this app 0 means customer is not a member of this app.

7) Loyalty points: It is of integer type. 0 means the person is not a member. Denotes the no of points the customer has.

8) E-wallet balance: It is of integer type. It contains the balance that customers possess in the e-wallet.

Restaurant Credentials:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Restauran	Restauran	X	Y	Cuisine	Menu	Username	Password	Ratings	No of time	offer	min_amount	
2	bg1	burgerking	2	3	burger	burgerking	burgerking	Whopper1	4.22	30	20	100	
3	bg2	burgerking	33	28	burger	burgerking	burgerking	Whopper1	4.72	13	20	100	
4	md1	mcdonalds	27	31	burger	mcdonalds	mcdkingfa	Whopper1	3.8	10	30	300	
5	md2	mcdonalds	8	8	burger	mcdonalds	mcdkingfa	Whopper1	3.7	9	30	300	
6	cb1	cheesyjuic	11	13	burger	cheesyjuic	cjb1243	cjbt124!	4.5	8	35	100	
7	cb2	cheesyjuic	24	24	burger	cheesyjuic	cjb124	cjbt124!	4.1	7	35	100	
8	bgk	burgerking	14	19	burger	burgerking	king05	@king	4.3	6	60	200	
9	dpz1	dominospi	2	16	pizza	dominos_	dominosfa	Dominos1	4.7	5	20	100	
0	dpz2	dominospi	33	8	pizza	dominos_	dominosfa	Dominos1	4.8	4	20	100	
1	pzh1	pizzahut1	27	23	pizza	pizzahut_n	pizzahutfa	PizzaHut12	4.9	7	35	300	
2	pzh2	pizzahut2	8	2	pizza	pizzahut_n	pizzahutfa	PizzaHut12	4.2	8	35	300	
3	pj1	papajohns	12	26	pizza	papajohns	papajohns	PapaJohns	4.1	9	40	250	
4	pj2	papajohns	19	11	pizza	papajohns	papajohns	PapaJohns	3.7	8	40	250	
5	cp	cheezypizz	24	14	pizza	cheezypizz	cheezypizz	CheezyPizz	4.5	6	50	150	
6	ms1	mansukhs1	3	8	North Indi	mansukhs_	mansukhsf	Mansukhs1	4.1	5	20	200	
7	ms2	mansukhs2	6	34	North Indi	mansukhs_	mansukhsf	Mansukhs1	3.6	7	20	200	
8	pd1	pind1	12	23	North Indi	pind_menu	pindfan123	Pind123!	3.9	7	30	150	
9	pd2	pind2	33	21	North Indi	pind_menu	pindfan124	Pind124!	4.3	8	30	150	
0	de1	delhi expre	27	14	North Indi	delhi_expr	delhifan12	Delhi123!	4.6	10	40	300	
1	de2	delhi expre	24	26	North Indi	delhi_expr	delhifan12	Delhi124!	4.5	11	40	300	
2	dc	dhaba cha	16	1	North Indi	dhaba_cha	dhabafan1	Dhaba123	4.7	12	20	100	
3	an1	al nassar1	1	13	Arabic	al_nassar_	alnassarfa	AlNassar1	4.4	9	25	200	
4	an2	al nassar2	34	1	Arabic	al_nassar_	alnassarfa	AlNassar1	4.2	7	25	200	
5	ah1	al hilal1	27	7	Arabic	al_hilal_m	alhilalfan1	AlHilal123!	4.6	6	30	300	
6	ah2	al hilal2	23	18	Arabic	al_hilal_m	alhilalfan1	AlHilal124!	4.8	8	30	300	
7	ak1	al kebabs1	18	33	Arabic	al_kebabs_	alkebabsfa	AlKebabs1	4.3	9	45	250	

1) Restaurant Id: it is of string type.

2) Restaurant Name: it is of string type.

3) X: It is of integer type and contains the X coordinate of the location of the restaurant.

4) Y: It is of integer type and contains the Y coordinate of the location of the restaurant.

5) Cuisine: it of string type.

6) Menu: Contains the csv file name where the menu is stored.

7) Username: It is of string type.

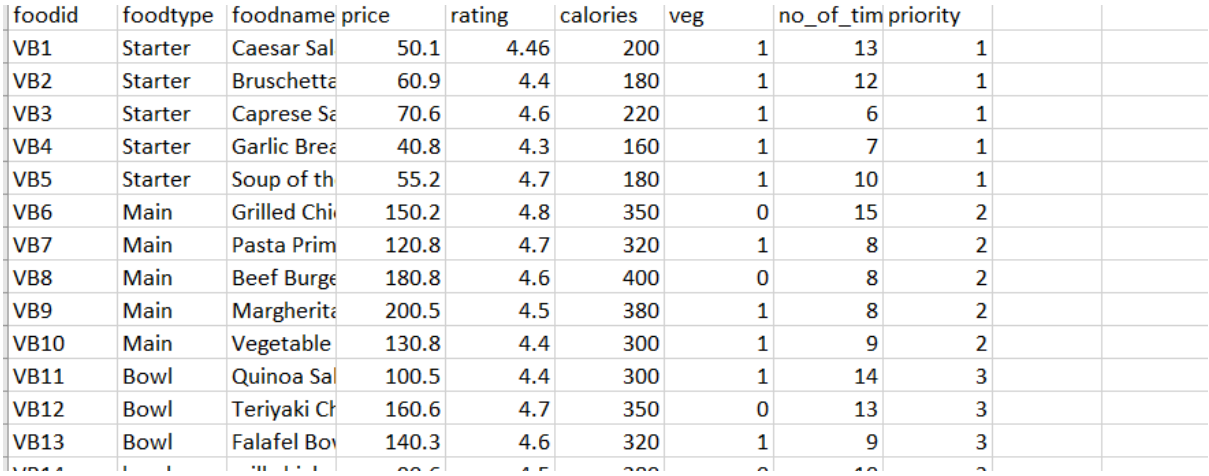
8) Password: It is of string type.

9) Ratings: It is of float type. Contains the overall ratings of the restaurant given by customers.

10) No of times ordered: It is of integer type. Gives the no of times people ordered from that restaurant

11) Min_amount: it is of integer type . It gives the maximum amount up to which discount offer can be applied on the bill.

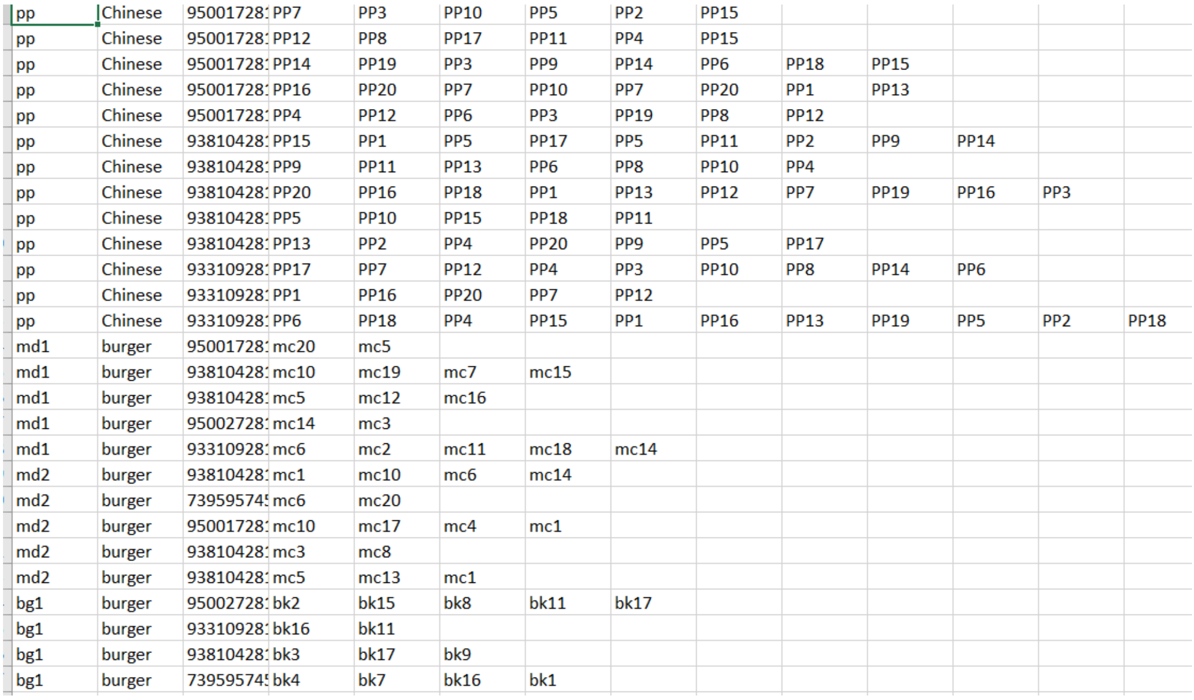
Menu for each restaurant:



foodid	foodtype	foodname	price	rating	calories	veg	no_of_tim	priority
VB1	Starter	Caesar Sal	50.1	4.46	200	1	13	1
VB2	Starter	Bruschetta	60.9	4.4	180	1	12	1
VB3	Starter	Caprese Sa	70.6	4.6	220	1	6	1
VB4	Starter	Garlic Brea	40.8	4.3	160	1	7	1
VB5	Starter	Soup of th	55.2	4.7	180	1	10	1
VB6	Main	Grilled Chi	150.2	4.8	350	0	15	2
VB7	Main	Pasta Prim	120.8	4.7	320	1	8	2
VB8	Main	Beef Burge	180.8	4.6	400	0	8	2
VB9	Main	Margherita	200.5	4.5	380	1	8	2
VB10	Main	Vegetable	130.8	4.4	300	1	9	2
VB11	Bowl	Quinoa Sal	100.5	4.4	300	1	14	3
VB12	Bowl	Teriyaki Ch	160.6	4.7	350	0	13	3
VB13	Bowl	Falafel Bo	140.3	4.6	320	1	9	3

1) Foodid: It is of string type.
2) Foodtype: It is of string type.
3) Foodname: It is of string type.
4) Price: It is of float type. Gives the price of one item of the food.
5) Calories: it is of integer type. Shown to customers so that they can be health conscious while ordering food.
6) Veg or Non Veg: 1 means it is veg. 0 means it is non veg.
7) No_of_times_ordered: it is of integer type. Refers to the no of times the food was ordered by the customers before.
8) Priority: Used for recommendation.

Previous Order Histories:



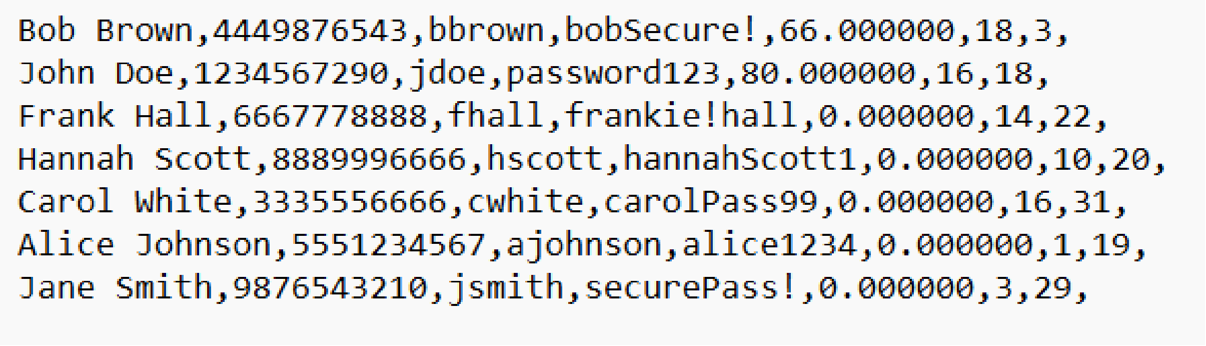
pp	Chinese	95001728	PP7	PP3	PP10	PP5	PP2	PP15					
pp	Chinese	95001728	PP12	PP8	PP17	PP11	PP4	PP15					
pp	Chinese	95001728	PP14	PP19	PP3	PP9	PP14	PP6	PP18	PP15			
pp	Chinese	95001728	PP16	PP20	PP7	PP10	PP7	PP20	PP1	PP13			
pp	Chinese	95001728	PP4	PP12	PP6	PP3	PP19	PP8	PP12				
pp	Chinese	93810428	PP15	PP1	PP5	PP17	PP5	PP11	PP2	PP9	PP14		
pp	Chinese	93810428	PP9	PP11	PP13	PP6	PP8	PP10	PP4				
pp	Chinese	93810428	PP20	PP16	PP18	PP1	PP13	PP12	PP7	PP19	PP16	PP3	
pp	Chinese	93810428	PP5	PP10	PP15	PP18	PP11						
pp	Chinese	93810428	PP13	PP2	PP4	PP20	PP9	PP5	PP17				
pp	Chinese	93310928	PP17	PP7	PP12	PP4	PP3	PP10	PP8	PP14	PP6		
pp	Chinese	93310928	PP1	PP16	PP20	PP7	PP12						
pp	Chinese	93310928	PP6	PP18	PP4	PP15	PP1	PP16	PP13	PP19	PP5	PP2	PP18
md1	burger	95001728	mc20	mc5									
md1	burger	93810428	mc10	mc19	mc7	mc15							
md1	burger	93810428	mc5	mc12	mc16								
md1	burger	95002728	mc14	mc3									
md1	burger	93310928	mc6	mc2	mc11	mc18	mc14						
md2	burger	93810428	mc1	mc10	mc6	mc14							
md2	burger	73959574	mc6	mc20									
md2	burger	95001728	mc10	mc17	mc4	mc1							
md2	burger	93810428	mc3	mc8									
md2	burger	93810428	mc5	mc13	mc1								
bg1	burger	95002728	bk2	bk15	bk8	bk11	bk17						
bg1	burger	93310928	bk16	bk11									
bg1	burger	93810428	bk3	bk17	bk9								
bg1	burger	73959574	bk4	bk7	bk16	bk1							

The columns present are:

1) Restaurant id: it is of string type.
2) Cuisine: it is of string type.
3) Customer id: it is of string type.
4) The items that the user has ordered in the form of food id.

The previous order histories database consists of the items which various users have ordered previously. Used for the food recommendation module.

Delivery driver details:



```
Bob Brown,4449876543,bbrown,bobSecure!,66.000000,18,3,
John Doe,1234567290,jdoe,password123,80.000000,16,18,
Frank Hall,6667778888,fhall,frankie!hall,0.000000,14,22,
Hannah Scott,8889996666,hscott,hannahScott1,0.000000,10,20,
Carol White,3335556666,cwhite,carolPass99,0.000000,16,31,
Alice Johnson,5551234567,ajohnson,alice1234,0.000000,1,19,
Jane Smith,9876543210,jsmith,securePass!,0.000000,3,29,
```

1) Driver name: char type
2) Phone number: char type
3) Username: char type
4) Password: char type
5) Bank balance: float type
6) x coordinate: int type
7) y coordinate: int type

Platform used for code development:

Code::Blocks platform was used for code development. Code::Blocks is an open-source integrated development environment (IDE) that's popular among developers for its flexibility and ease of use, especially for C, C++, and Fortran programming languages. It supports multiple compilers and offers features like syntax highlighting, code completion, and debugging tools, making it suitable for both beginners and advanced programmers. Its modular architecture allows users to add plugins to extend functionality ,enhancing productivity and customization. Overall, Code::Blocks is valued for its simplicity, robustness, and active community support.

Language Constructs used in modules:

In the food delivery system various language constructs have been used they are:

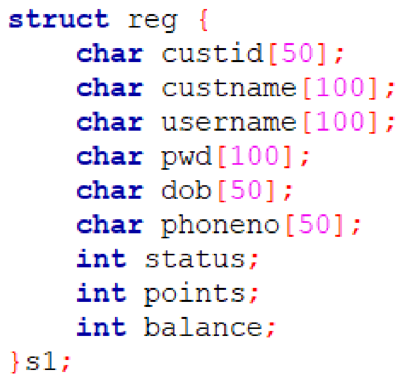
Array
Structures

Array of structures
Files

Files: Files are used so as to ensure that the large amount of data is stored permanently and we can use the data anytime. It is also used for sharing data to multiple modules. We used CSV files as it helps in faster access towards data and data handling is a lot easier.

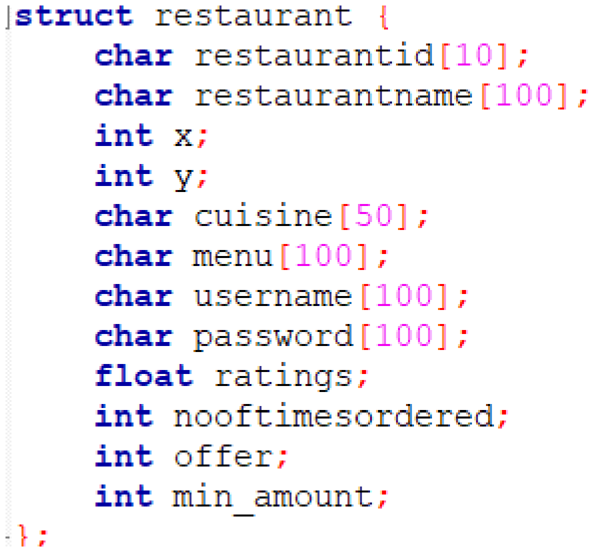
Structures: Structures are used in various modules so that we can store data of different types under a single construct. They also handle memory better and they are very useful when dealing with file handling. It is also useful for parsing complex data to the functions. Various structures were used in handling different data types, they are:

1) Struct reg is a structure which is used in order to extract the customer credentials from the customer credentials file.



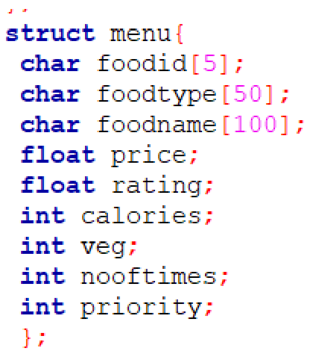
```
struct reg {
    char custid[50];
    char custname[100];
    char username[100];
    char pwd[100];
    char dob[50];
    char phoneno[50];
    int status;
    int points;
    int balance;
}s1;
```

2) Struct restaurant is a structure which is used in order to extract the restaurant credentials from the restaurant credentials file.



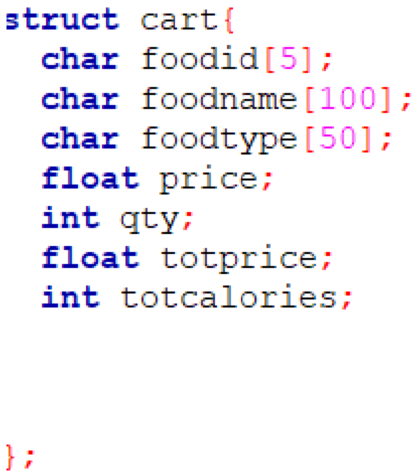
```
struct restaurant {
    char restaurantid[10];
    char restaurantname[100];
    int x;
    int y;
    char cuisine[50];
    char menu[100];
    char username[100];
    char password[100];
    float ratings;
    int nooftimesordered;
    int offer;
    int min_amount;
};
```

3) Struct menu is a structure which is used in order to extract the menu from files which contain the menu of the restaurants.



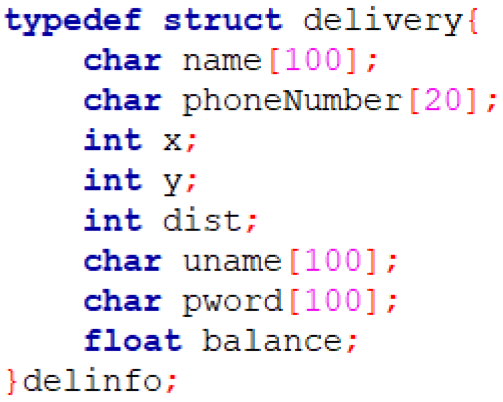
```
struct menu{
 char foodid[5];
 char foodtype[50];
 char foodname[100];
 float price;
 float rating;
 int calories;
 int veg;
 int nooftimes;
 int priority;
 };
```

4) Struct cart is used in order to store the cart items which the user ordered during the item selection process.



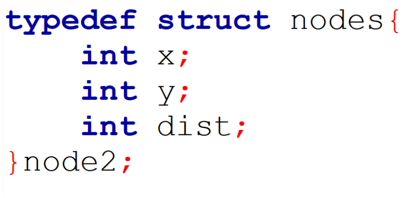
```
struct cart{
  char foodid[5];
  char foodname[100];
  char foodtype[50];
  float price;
  int qty;
  float totprice;
  int totcalories;



};
```

5) Struct delivery is used in order to store the driver details from the driver details database.



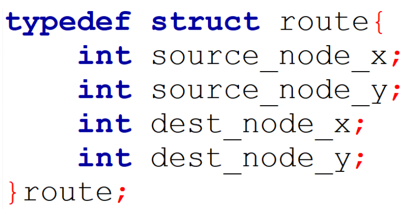
```
typedef struct delivery{
     char name[100];
     char phoneNumber[20];
     int x;
     int y;
     int dist;
     char uname[100];
     char pword[100];
     float balance;
}delinfo;
```

6) Struct node is used to find the surrounding nodes of the given coordinate.



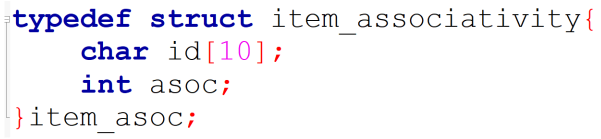
```
typedef struct nodes{
     int x;
     int y;
     int dist;
}node2;
```

7) Struct route is used to store the two nodes which connect the two-coordinates in map via the shortest path.



```
typedef struct route{
    int source_node_x;
    int source_node_y;
    int dest_node_x;
    int dest_node_y;
}route;
```

8) This structure is used to store the food-id and its respective associativity (with respect to current items in cart) with reference to the previous orders.



```
typedef struct item_associativity{
    char id[10];
    int asoc;
}item_asoc;
```

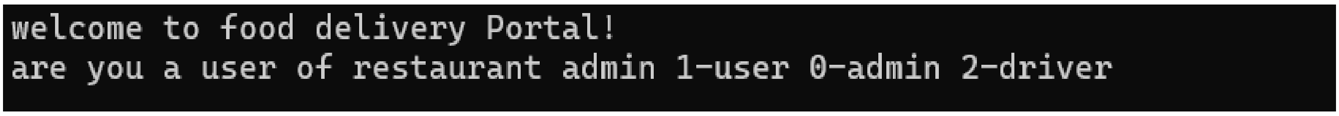
Array of structures: An array of structures is used in programming to store multiple instances of a structured data type in contiguous memory locations.Array of structures are useful in keeping related data together and easy access of data.Also array of structures can easily be passed to functions through pass by reference which helps in modification of data a lot easier. Array of structures used in food delivery system are:

1) struct reg custlog[]: used to store data of the customer credentials in the form of an array of structures where each structure contains the credentials of a customer.
2) struct menu arr1[]: used to store data of the menu items of a restaurant in the form of an array of structures where each structure contains a menu item description.
3) struct cart cart1[]: used to store data of cart items of the user in the form of an array of structures where each structure represents an item the user has ordered.
4) delinfo drivers[15] : used to store the data of driver details in the form of an array of structures where each structure represents a driver detail.
5) Item_asoc assocs[]: Used to store all the food-ids available in the menu and to hold their respective associative points.

Array: An array is used in order to hold values of the same data type in contiguous memory locations. In the food delivery system all modules use character arrays (strings) and arrays of various other data types.

VALIDATION OF PROPOSED SOLUTION:

The food delivery system asks whether the given person is a user, restaurant admin or a driver. If the person is a user he enters 1, if he is an admin he enters 0, if he is a driver he enters 2.



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin 2-driver
```

Customer related modules:

Customer Login module:

Input: the user is given 3 options:

1- register

2-login

3- forgot password

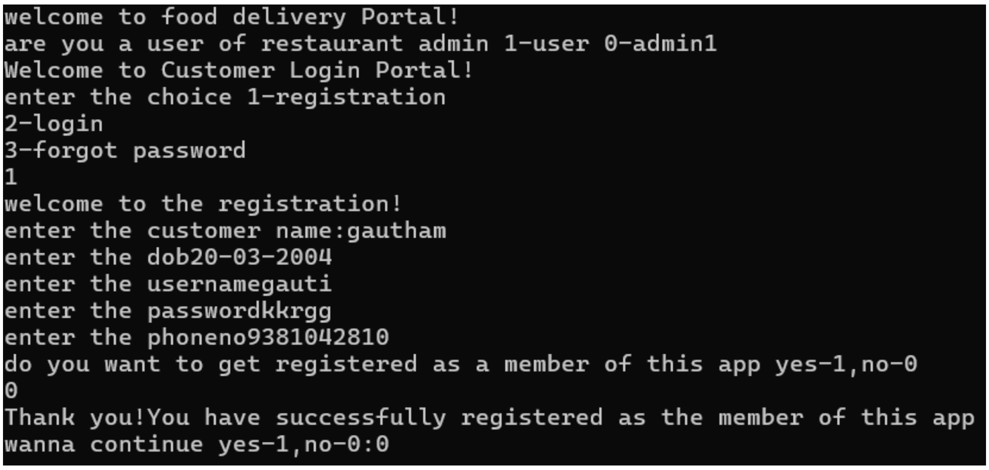
Registration:

Input: The user is asked to give details such as name,dob, phone number, username,password. He is also asked whether he wants to get registered as a member of the app. If not, the number of membership points is set to zero.

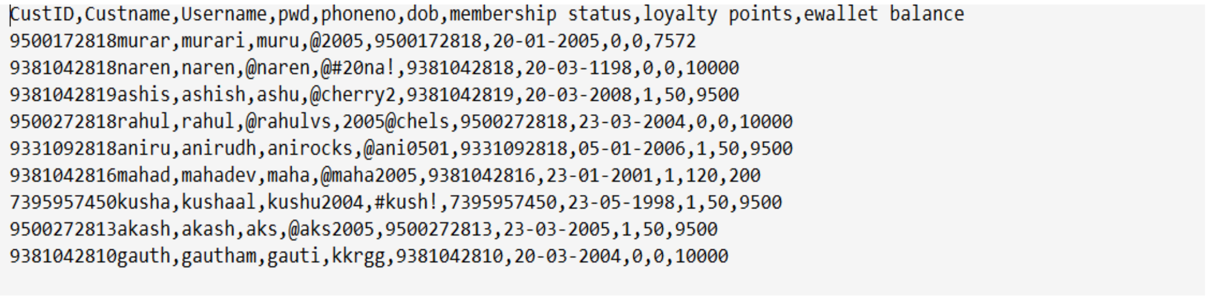
Expected Output: membership status is given 0

No of points becomes 0..

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin1
Welcome to Customer Login Portal!
enter the choice 1-registration
2-login
3-forgot password
1
welcome to the registration!
enter the customer name:gautham
enter the dob20-03-2004
enter the usernamegauti
enter the passwordkkrgg
enter the phoneno9381042810
do you want to get registered as a member of this app yes-1,no-0
0
Thank you!You have successfully registered as the member of this app
wanna continue yes-1,no-0:0
```



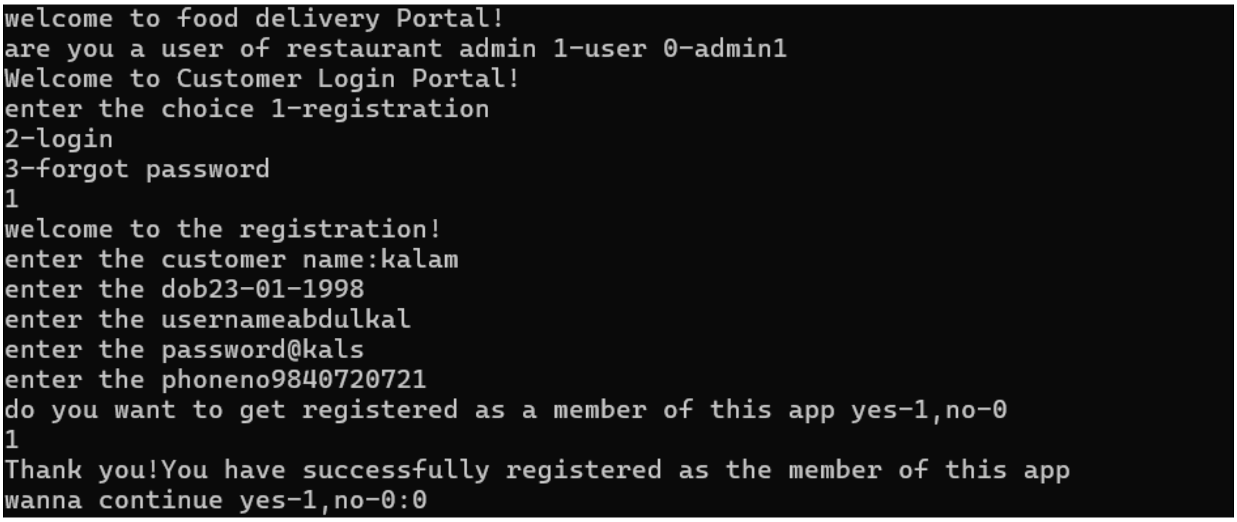
```
CustID,Custname,Username,pwd,phoneno,dob,membership status,loyalty points,ewallet balance
9500172818murar,murari,muru,@2005,9500172818,20-01-2005,0,0,7572
9381042818naren,naren,@naren,@#20na!,9381042818,20-03-1198,0,0,10000
9381042819ashis,ashish,ashu,@cherry2,9381042819,20-03-2008,1,50,9500
9500272818rahul,rahul,@rahulvs,2005@chels,9500272818,23-03-2004,0,0,10000
9331092818aniru,anirudh,anirocks,@ani0501,9331092818,05-01-2006,1,50,9500
9381042816mahad,mahadev,maha,@maha2005,9381042816,23-01-2001,1,120,200
7395957450kusha,kushaal,kushu2004,#kush!,7395957450,23-05-1998,1,50,9500
9500272813akash,akash,aks,@aks2005,9500272813,23-03-2005,1,50,9500
9381042810gauth,gautham,gauti,kkrgg,9381042810,20-03-2004,0,0,10000
```

Input: If he gives yes

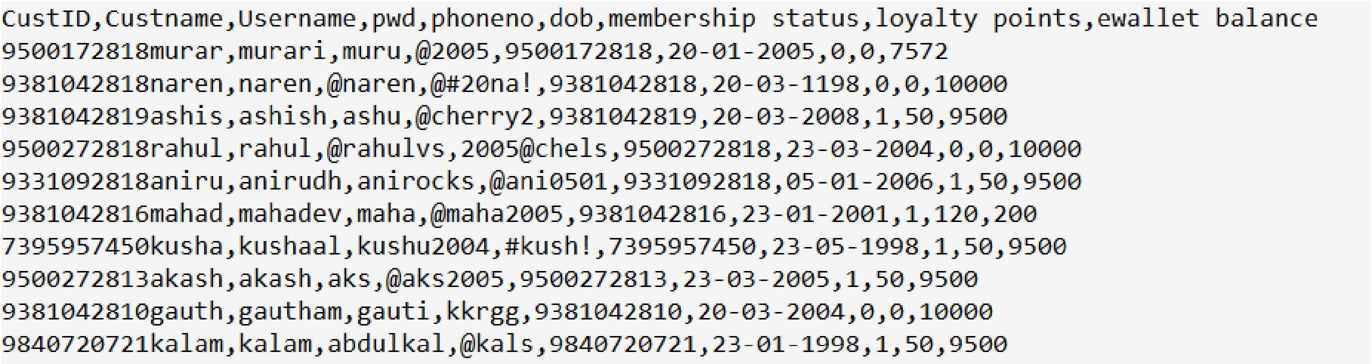
Expected Output: membership status

1. 500 is deducted from the e-wallet. 50 points is added to the points column.

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin1
Welcome to Customer Login Portal!
enter the choice 1-registration
2-login
3-forgot password
1
welcome to the registration!
enter the customer name:kalam
enter the dob23-01-1998
enter the usernameabdulkal
enter the password@kals
enter the phoneno9840720721
do you want to get registered as a member of this app yes-1,no-0
1
Thank you!You have successfully registered as the member of this app
wanna continue yes-1,no-0:0
```



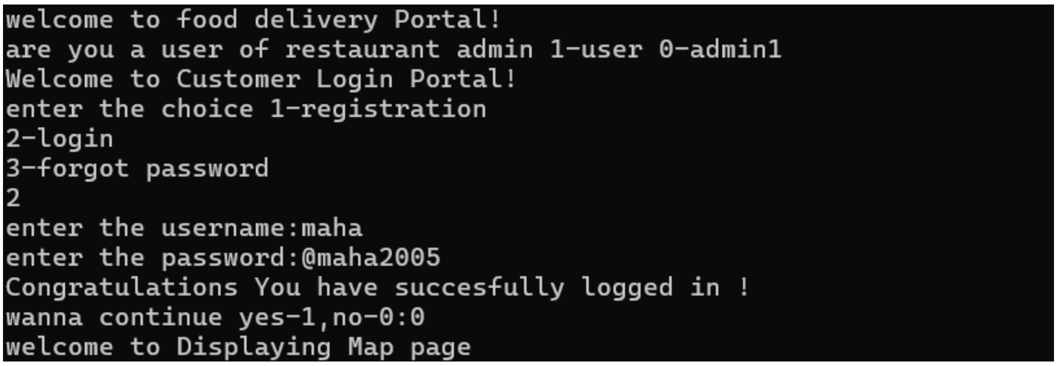
```
CustID,Custname,Username,pwd,phoneno,dob,membership status,loyalty points,ewallet balance
9500172818murar,murari,muru,@2005,9500172818,20-01-2005,0,0,7572
9381042818naren,naren,@naren,@#20na!,9381042818,20-03-1198,0,0,10000
9381042819ashis,ashish,ashu,@cherry2,9381042819,20-03-2008,1,50,9500
9500272818rahul,rahul,@rahulvs,2005@chels,9500272818,23-03-2004,0,0,10000
9331092818aniru,anirudh,anirocks,@ani0501,9331092818,05-01-2006,1,50,9500
9381042816mahad,mahadev,maha,@maha2005,9381042816,23-01-2001,1,120,200
7395957450kusha,kushaal,kushu2004,#kush!,7395957450,23-05-1998,1,50,9500
9500272813akash,akash,aks,@aks2005,9500272813,23-03-2005,1,50,9500
9381042810gauth,gautham,gauti,kkrgg,9381042810,20-03-2004,0,0,10000
9840720721kalam,kalam,abdulkal,@kals,9840720721,23-01-1998,1,50,9500
```

Login:

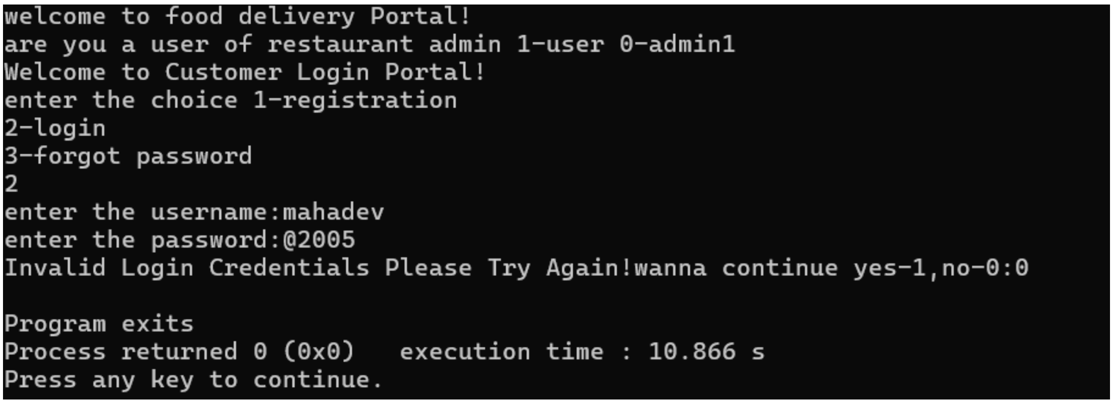
Input: enter username, password

Expected output: gives login successful if the credentials are correct and proceeds to the next module else it states that the credentials are incorrect.

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin1
Welcome to Customer Login Portal!
enter the choice 1-registration
2-login
3-forgot password
2
enter the username:maha
enter the password:@maha2005
Congratulations You have succesfully logged in !
wanna continue yes-1,no-0:0
welcome to Displaying Map page
```



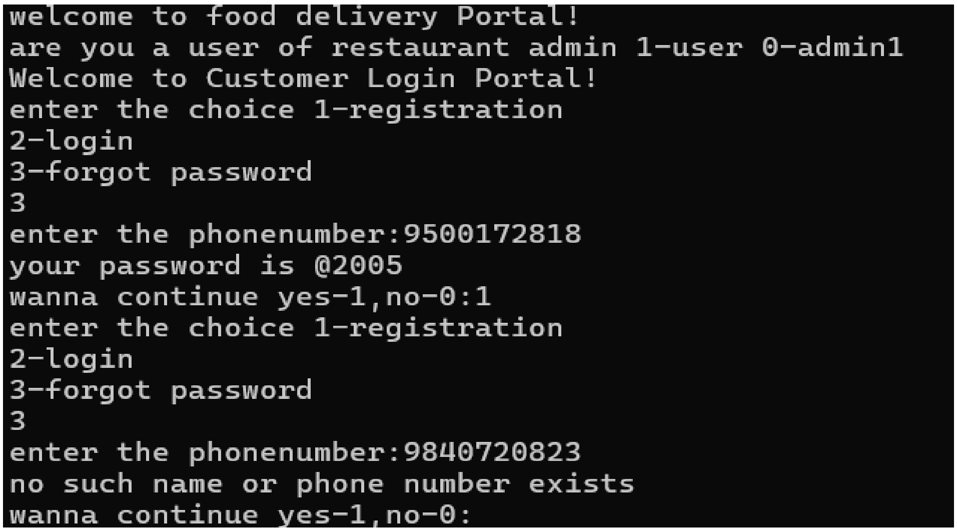
```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin1
Welcome to Customer Login Portal!
enter the choice 1-registration
2-login
3-forgot password
2
enter the username:mahadev
enter the password:@2005
Invalid Login Credentials Please Try Again!wanna continue yes-1,no-0:0

Program exits
Process returned 0 (0x0)   execution time : 10.866 s
Press any key to continue.
```

Forgot Password:

Input: to give the phone number of the person

Expected Output: returns the password based on the phone number if it exists. Else, says no such phone number exists.

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin1
Welcome to Customer Login Portal!
enter the choice 1-registration
2-login
3-forgot password
3
enter the phonenumber:9500172818
your password is @2005
wanna continue yes-1,no-0:1
enter the choice 1-registration
2-login
3-forgot password
3
enter the phonenumber:9840720823
no such name or phone number exists
wanna continue yes-1,no-0:
```

Item Selection module:

Item selection module involves various steps:

1) Entering delivery address in the form of x and y coordinates and the cuisine.

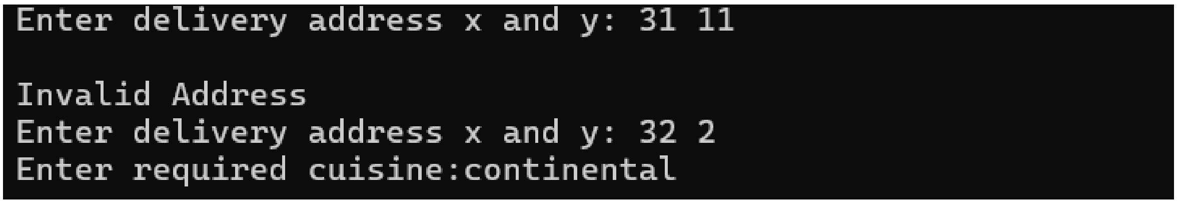
2) Ranking the restaurants based on certain parameters.
3) Searching for a specific food item.

Entering delivery address:

Input: enter x and y coordinates.

Expected output: proceeds to ranking if they do not coincide with the location of roads or the restaurants. Else asks the user to enter the coordinates again.

Output:



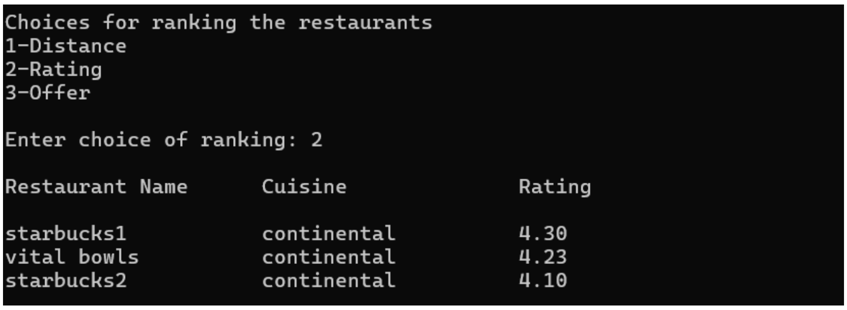
```
Enter delivery address x and y: 31 11

Invalid Address
Enter delivery address x and y: 32 2
Enter required cuisine:continental
```

Ranking the restaurants:

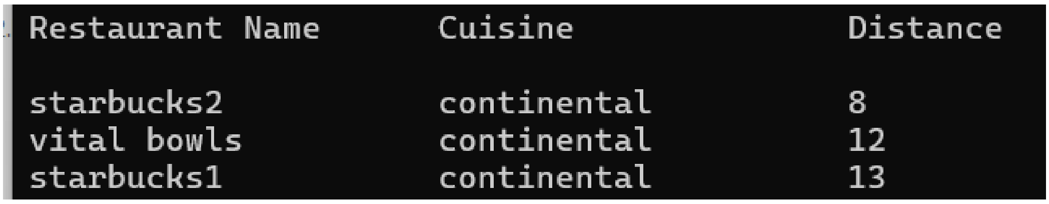
Input: choose whether to rank by distance, offers or ratings.

Expected Output: Gives the rankings based on the parameters in ascending or descending order.

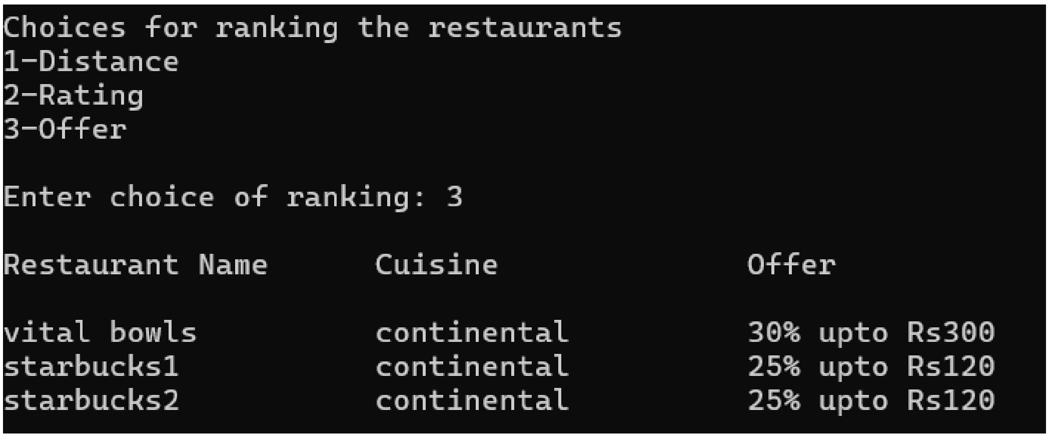
Output:



```
Choices for ranking the restaurants
1-Distance
2-Rating
3-Offer

Enter choice of ranking: 2

Restaurant Name      Cuisine              Rating

starbucks1           continental          4.30
vital bowls          continental          4.23
starbucks2           continental          4.10
```



```
Restaurant Name      Cuisine              Distance

starbucks2           continental          8
vital bowls          continental          12
starbucks1           continental          13
```



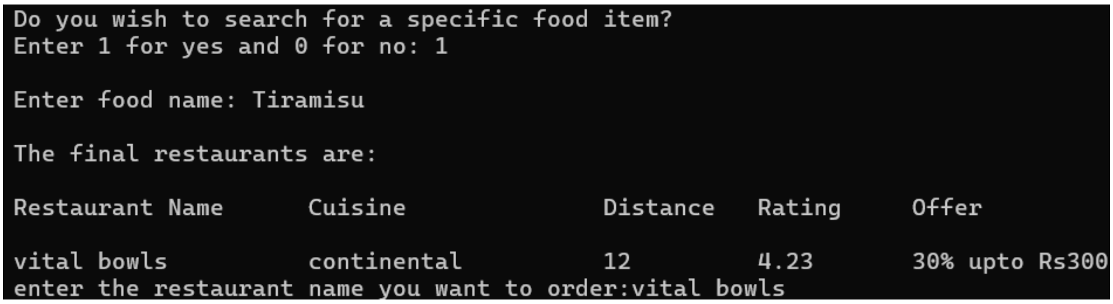
```
Choices for ranking the restaurants
1-Distance
2-Rating
3-Offer

Enter choice of ranking: 3

Restaurant Name      Cuisine              Offer

vital bowls          continental          30% upto Rs300
starbucks1           continental          25% upto Rs120
starbucks2           continental          25% upto Rs120
```

Searching for a specific food item:

Input: It asks the user to enter whether he wants to enter a specific food item or not. If so, mention the food item.

Expected Output: gives the restaurant which has the food item among the filtered restaurants.

Output:



```
Do you wish to search for a specific food item?
Enter 1 for yes and 0 for no: 1

Enter food name: Tiramisu

The final restaurants are:

Restaurant Name      Cuisine              Distance   Rating     Offer

vital bowls          continental          12         4.23       30% upto Rs300
enter the restaurant name you want to order:vital bowls
```

Cart item module:

In the cartitem module there are 4 options, they are:

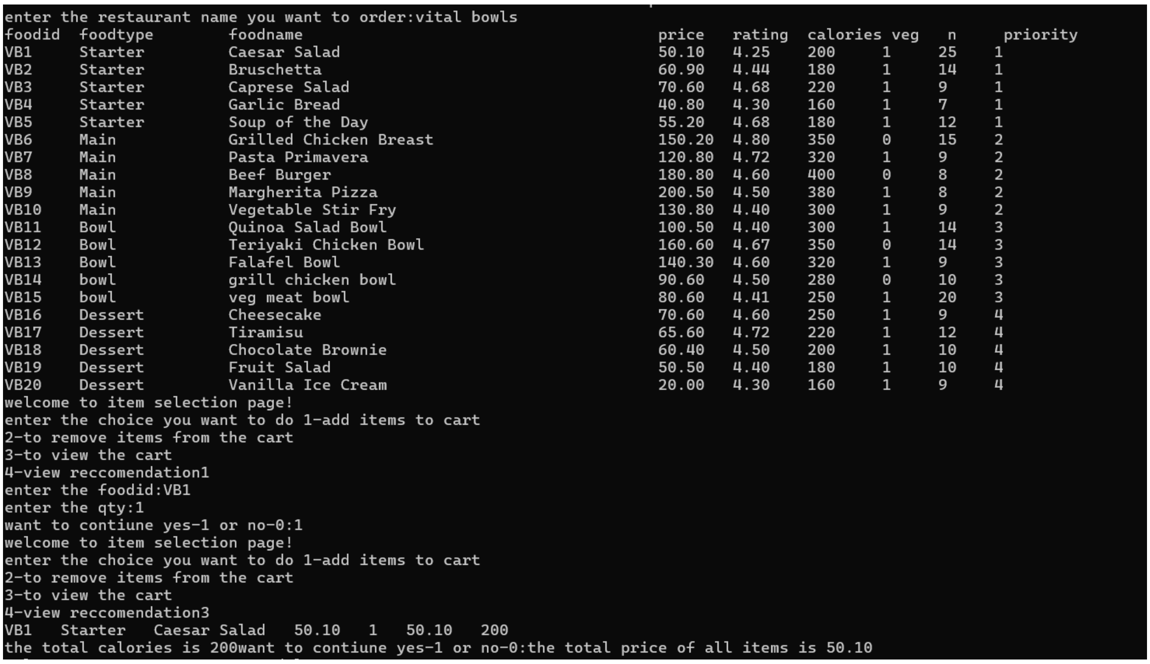
1) Adding items to the cart.
2) Removing items from the cart.
3) View the cart.
4) Recommendation.

Adding items to the cart:

Input: enter the food id, quantity.

Expected Output: the item will get added to the cart.

Output:



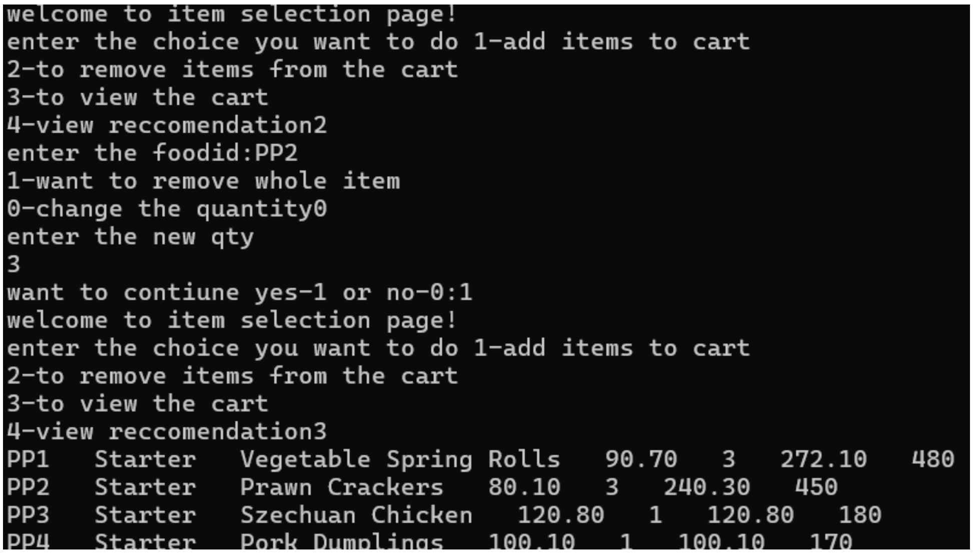
```
enter the restaurant name you want to order:vital bowls
foodid  foodtype        foodname                                        price   rating  calories veg   n     priority
VB1     Starter         Caesar Salad                                    50.10   4.25    200      1     25    1
VB2     Starter         Bruschetta                                      60.90   4.44    180      1     14    1
VB3     Starter         Caprese Salad                                   70.60   4.68    220      1     9     1
VB4     Starter         Garlic Bread                                    40.80   4.30    160      1     7     1
VB5     Starter         Soup of the Day                                 55.20   4.68    180      1     12    1
VB6     Main            Grilled Chicken Breast                          150.20  4.80    350      0     15    2
VB7     Main            Pasta Primavera                                 120.80  4.72    320      1     9     2
VB8     Main            Beef Burger                                     180.80  4.60    400      0     8     2
VB9     Main            Margherita Pizza                                200.50  4.50    380      1     8     2
VB10    Main            Vegetable Stir Fry                              130.80  4.40    300      1     9     2
VB11    Bowl            Quinoa Salad Bowl                               100.50  4.40    300      1     14    3
VB12    Bowl            Teriyaki Chicken Bowl                           160.60  4.67    350      0     14    3
VB13    Bowl            Falafel Bowl                                    140.30  4.60    320      1     9     3
VB14    bowl            grill chicken bowl                              90.60   4.50    280      0     10    3
VB15    bowl            veg meat bowl                                   80.60   4.41    250      1     20    3
VB16    Dessert         Cheesecake                                      70.60   4.60    250      1     9     4
VB17    Dessert         Tiramisu                                        65.60   4.72    220      1     12    4
VB18    Dessert         Chocolate Brownie                               60.40   4.50    200      1     10    4
VB19    Dessert         Fruit Salad                                     50.50   4.40    180      1     10    4
VB20    Dessert         Vanilla Ice Cream                               20.00   4.30    160      1     9     4
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation1
enter the foodid:VB1
enter the qty:1
want to contiune yes-1 or no-0:1
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation3
VB1   Starter   Caesar Salad   50.10   1   50.10   200
the total calories is 200want to contiune yes-1 or no-0:the total price of all items is 50.10
```

Removing items from the cart:

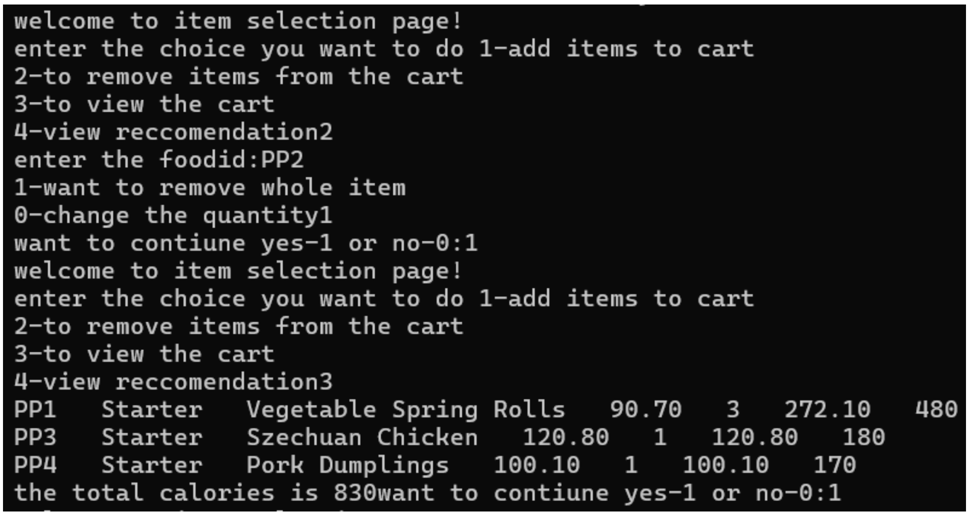
Input: Enter the food id to be removed. also mention whether you want to update quantity or remove the item as a whole

Expected Output: whole item getting removed as a whole or quantity gets updated

Output:



```
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation2
enter the foodid:PP2
1-want to remove whole item
0-change the quantity0
enter the new qty
3
want to contiune yes-1 or no-0:1
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation3
PP1   Starter   Vegetable Spring Rolls   90.70   3   272.10   480
PP2   Starter   Prawn Crackers   80.10   3   240.30   450
PP3   Starter   Szechuan Chicken   120.80   1   120.80   180
PP4   Starter   Pork Dumplings   100.10   1   100.10   170
```



```
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation2
enter the foodid:PP2
1-want to remove whole item
0-change the quantity1
want to contiune yes-1 or no-0:1
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation3
PP1   Starter   Vegetable Spring Rolls   90.70   3   272.10   480
PP3   Starter   Szechuan Chicken   120.80   1   120.80   180
PP4   Starter   Pork Dumplings   100.10   1   100.10   170
the total calories is 830want to contiune yes-1 or no-0:1
```

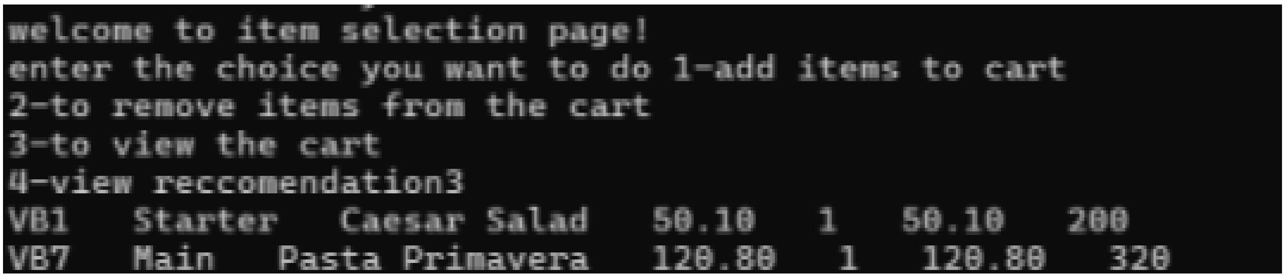
View Cart:

View cart is a module used for viewing the food items present in the cart.

Input: enter 3

Expected Output: returns the cart consisting of food items which have been ordered till now

Output:

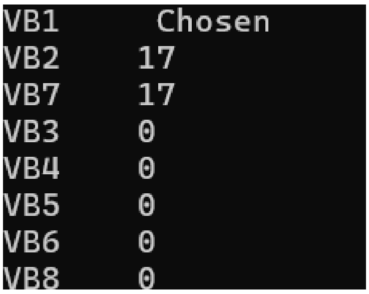


```
welcome to item selection page!
enter the choice you want to do 1-add items to cart
2-to remove items from the cart
3-to view the cart
4-view reccomendation3
VB1   Starter   Caesar Salad   50.10   1   50.10   200
VB7   Main   Pasta Primavera   120.80   1   120.80   320
```

Food Recommendation:

It gives recommendations based on previous order histories by various users on that particular restaurant.

Output: gives recommended food items which are not ordered.



```
VB1      Chosen
VB2     17
VB7     17
VB3     0
VB4     0
VB5     0
VB6     0
VB8     0
```

Here VB1 is chosen already and gives other items which can be ordered.

Payment Module:

Payment module consists of 2 parts

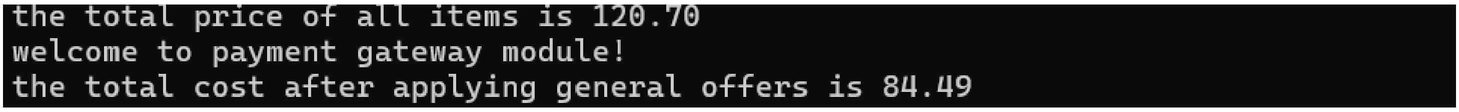
1) Total cost after applying general offers
2) Total cost after applying membership offers

Total cost after applying general offers:

Input: None.

Expected Output: returns the cost after applying general offers.

Output:



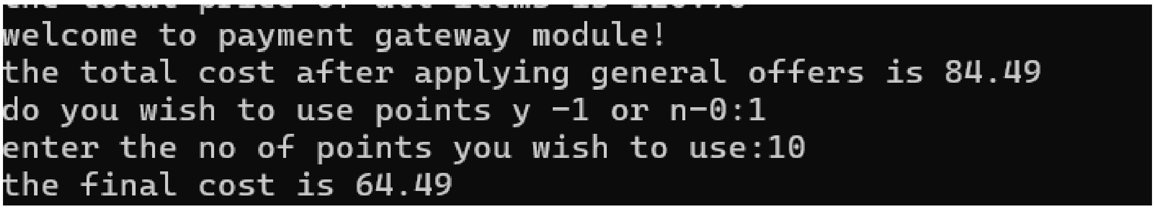
```
the total price of all items is 120.70
welcome to payment gateway module!
the total cost after applying general offers is 84.49
```

Total cost after applying membership offers:

Input: in case the user is a member it asks whether the user wants to use points or not and if so the number of points to be used.

Expected Output :points causing further deduction of total cost

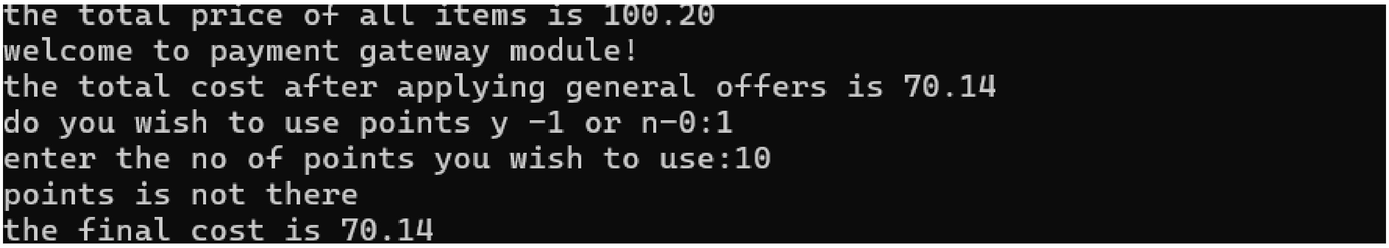
Case 1: If the user is a member and has enough points and bank balance.



```
welcome to payment gateway module!
the total cost after applying general offers is 84.49
do you wish to use points y -1 or n-0:1
enter the no of points you wish to use:10
the final cost is 64.49
```

Also the points get added to the points column in the customer database and the amount gets deducted from the e-wallet.

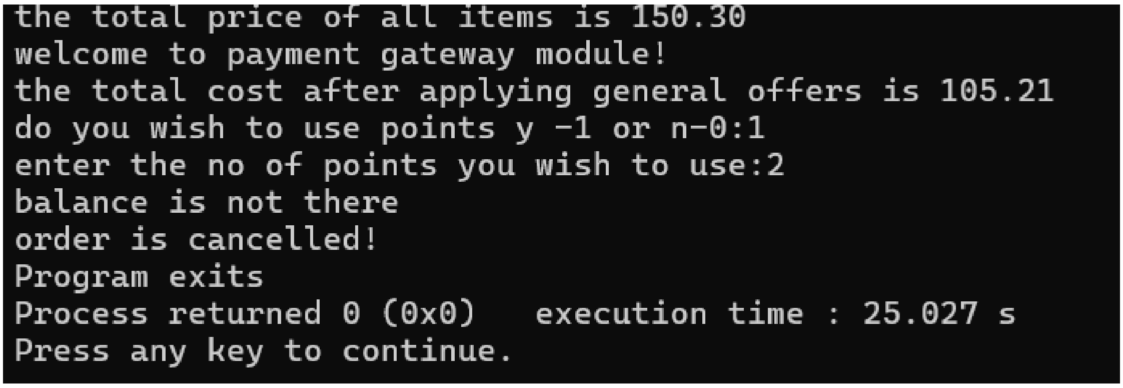
Case 2: If the user is a member and does not have enough points but has enough bank balance.



```
the total price of all items is 100.20
welcome to payment gateway module!
the total cost after applying general offers is 70.14
do you wish to use points y -1 or n-0:1
enter the no of points you wish to use:10
points is not there
the final cost is 70.14
```

Here the amount gets deducted from the e-wallet and the points get added.

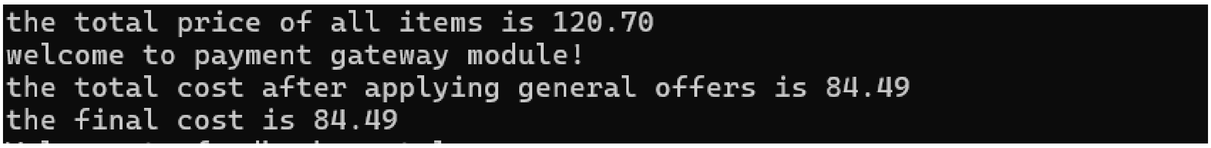
Case 3: If the user is a member has enough points but lacks bank balance.



```
the total price of all items is 150.30
welcome to payment gateway module!
the total cost after applying general offers is 105.21
do you wish to use points y -1 or n-0:1
enter the no of points you wish to use:2
balance is not there
order is cancelled!
Program exits
Process returned 0 (0x0)   execution time : 25.027 s
Press any key to continue.
```

Program ends since there is not enough balance to pay.

Case 4: If the user is not a member but has enough bank balance.



```
the total price of all items is 120.70
welcome to payment gateway module!
the total cost after applying general offers is 84.49
the final cost is 84.49
```

Only general offers are applied and the amount is deducted from the e-wallet.

Feedback Module:

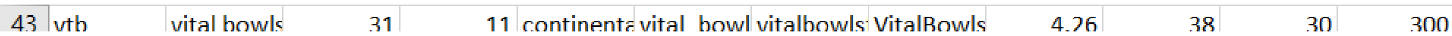
Feedback module is used to give ratings to the restaurant as well as individual food items which are ordered

1) Ratings to the restaurant

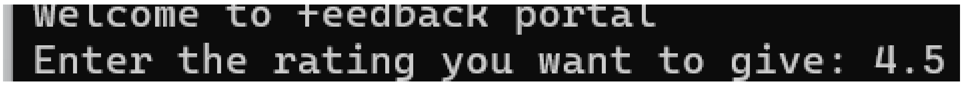
Input: the user is asked to give ratings out of 5

Expected output: the overall ratings get changed as a result of the new rating given by the user also the no of times ordered also increments by 1

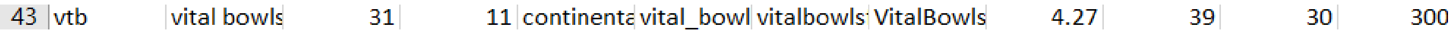
Output:



43	vtb	vital bowls	31	11	continenta	vital_bowl	vitalbowls	VitalBowls	4.26	38	30	300



```
Welcome to feedback portal
Enter the rating you want to give: 4.5
```



43	vtb	vital bowls	31	11	continenta	vital_bowl	vitalbowls	VitalBowls	4.27	39	30	300

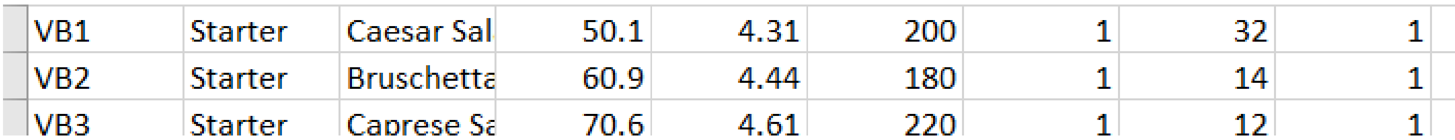
Here the ratings get changed and the no of times ordered get updated by 1.

2) Ratings for the food items

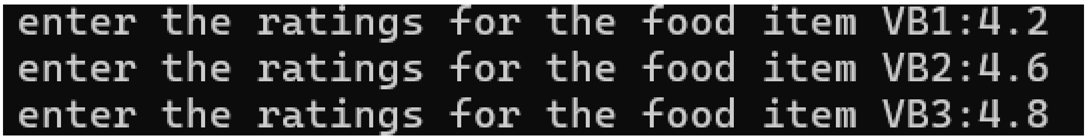
Input: ratings out of 5 have to be given to the individual food items.

Expected Output: ratings get updated in the menu database for the food items ordered and the no of times ordered also gets incremented by 1.

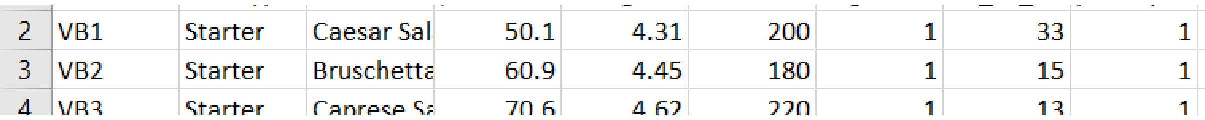
Output:



VB1	Starter	Caesar Sal	50.1	4.31	200	1	32	1
VB2	Starter	Bruschetta	60.9	4.44	180	1	14	1
VB3	Starter	Caprese Sa	70.6	4.61	220	1	12	1



```
enter the ratings for the food item VB1:4.2
enter the ratings for the food item VB2:4.6
enter the ratings for the food item VB3:4.8
```



2	VB1	Starter	Caesar Sal	50.1	4.31	200	1	33	1
3	VB2	Starter	Bruschetta	60.9	4.45	180	1	15	1
4	VB3	Starter	Caprese Sa	70.6	4.62	220	1	13	1

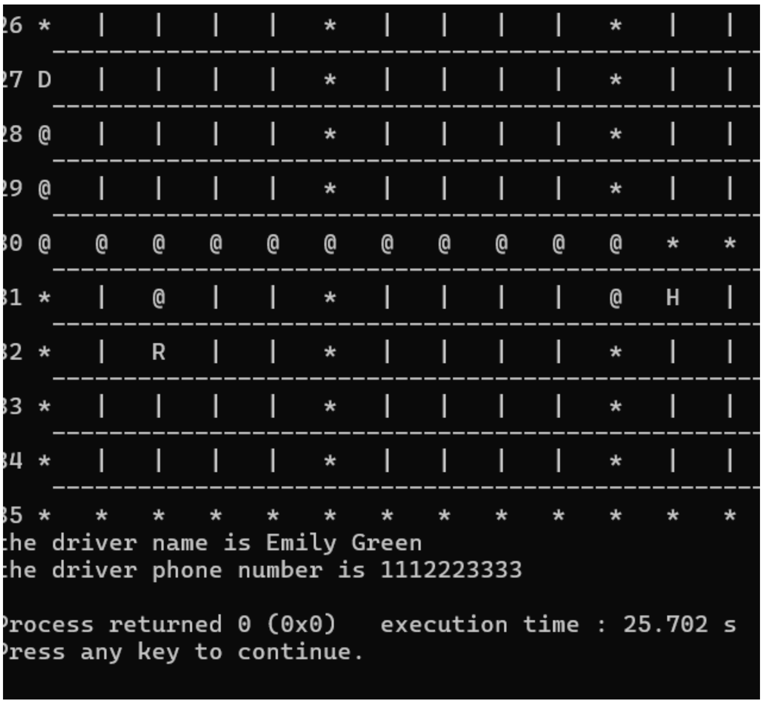
Driver Assigning Module:

In this module a driver is assigned to carry out the delivery process.

Input: None.

Expected Output: Displays the driver name and phone number and also a map showing the position of the driver.

Output:



```
26 *   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
27 D   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
28 @   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
29 @   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
30 @   @   @   @   @   @   @   @   @   @   @   *   *
     ------------------------------------------------
31 *   |   @   |   |   *   |   |   |   |   @   H   |
     ------------------------------------------------
32 *   |   R   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
33 *   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
34 *   |   |   |   |   *   |   |   |   |   *   |   |
     ------------------------------------------------
35 *   *   *   *   *   *   *   *   *   *   *   *   *
the driver name is Emily Green
the driver phone number is 1112223333

Process returned 0 (0x0)   execution time : 25.702 s
Press any key to continue.
```

Restaurant related modules:

1) Restaurant login
2) Menu alter

Restaurant login module:

Restaurant login module is for restaurant admins to login into their accounts.

There are 2 parts in the restaurant login module. They are:

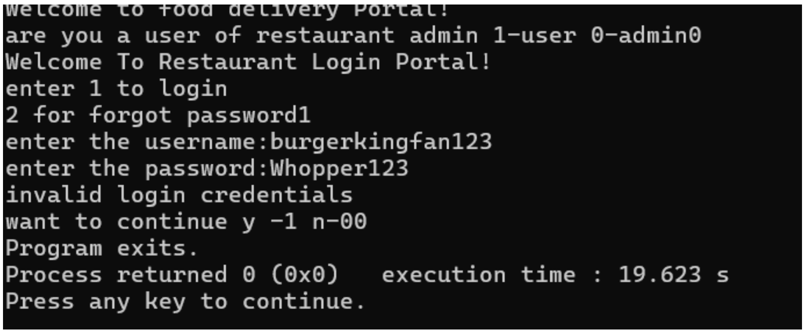
1) Login module
2) Forgot password

Login module:

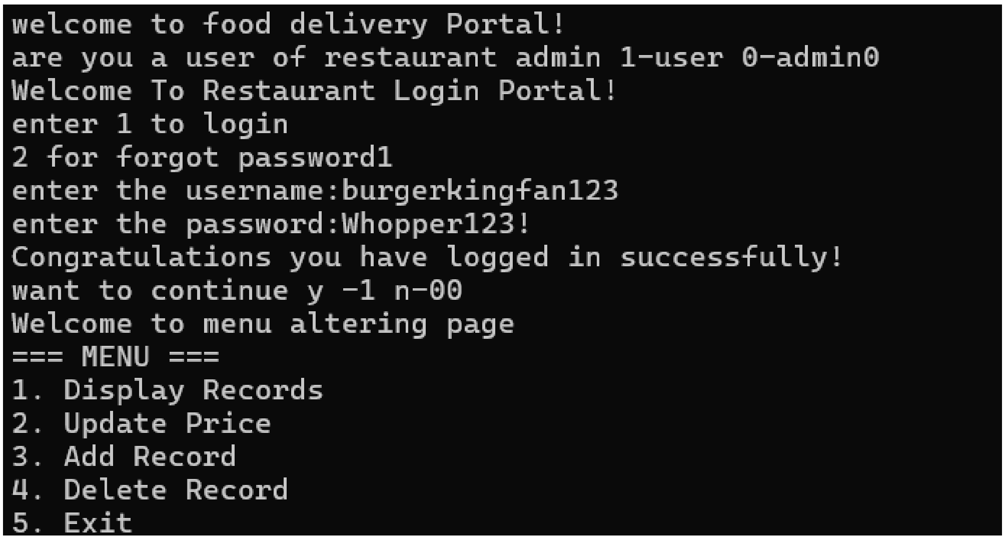
Input: Enter the username and password.

Expected Output : proceeds to the next module if the credentials are correct else says invalid credentials.

Output:



```
Welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin0
Welcome To Restaurant Login Portal!
enter 1 to login
2 for forgot password1
enter the username:burgerkingfan123
enter the password:Whopper123
invalid login credentials
want to continue y -1 n-00
Program exits.
Process returned 0 (0x0)   execution time : 19.623 s
Press any key to continue.
```



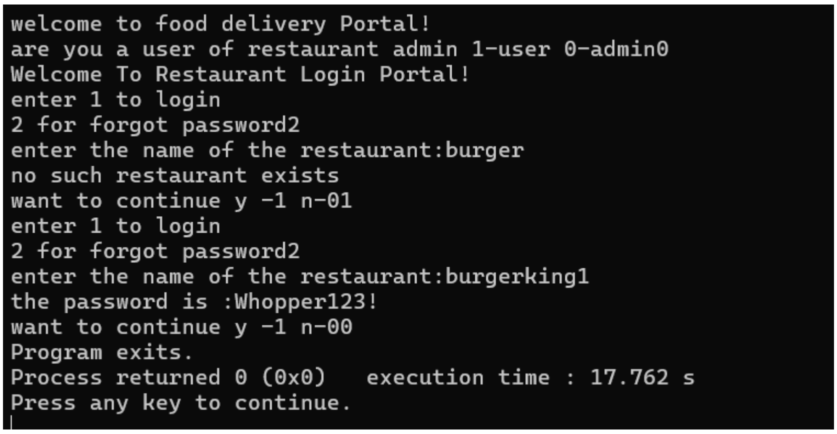
```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin0
Welcome To Restaurant Login Portal!
enter 1 to login
2 for forgot password1
enter the username:burgerkingfan123
enter the password:Whopper123!
Congratulations you have logged in successfully!
want to continue y -1 n-00
Welcome to menu altering page
=== MENU ===
1. Display Records
2. Update Price
3. Add Record
4. Delete Record
5. Exit
```

Forgot Password:

Input: Enter the name of the restaurant.

Expected Output: returns the password of the restaurant if name exists else says no such restaurant name exists.

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin0
Welcome To Restaurant Login Portal!
enter 1 to login
2 for forgot password2
enter the name of the restaurant:burger
no such restaurant exists
want to continue y -1 n-01
enter 1 to login
2 for forgot password2
enter the name of the restaurant:burgerking1
the password is :Whopper123!
want to continue y -1 n-00
Program exits.
Process returned 0 (0x0)   execution time : 17.762 s
Press any key to continue.
```

Menu Alter module:

Menu alter module is for the restaurant admin to change the menu of the restaurant. It consists of the following parts:

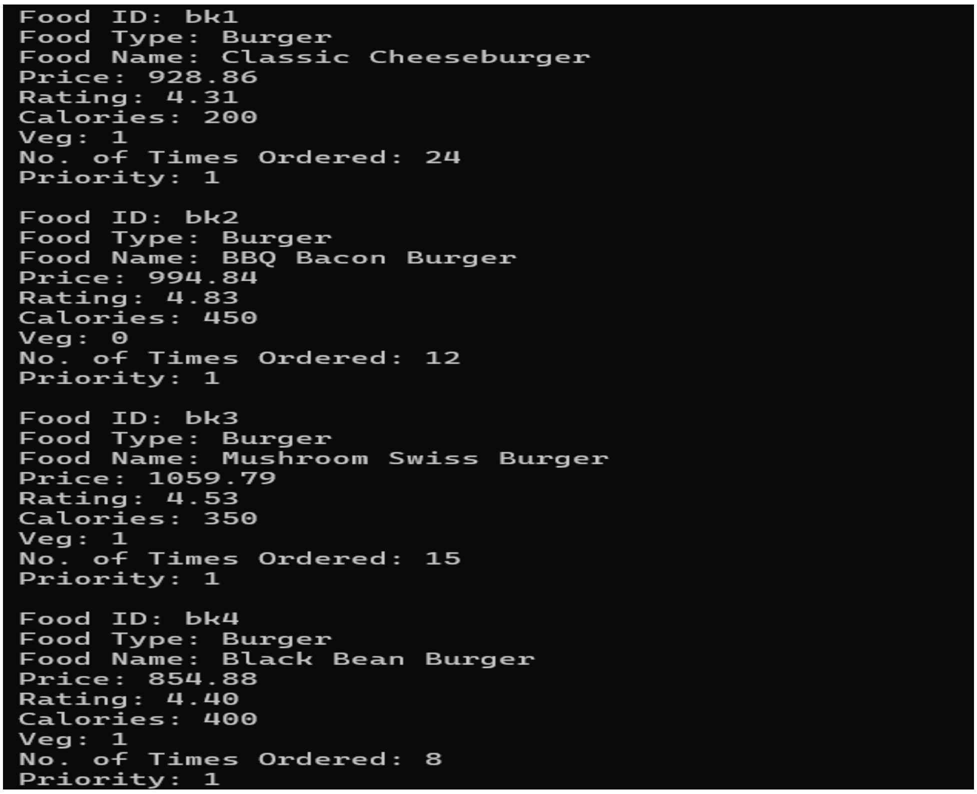
1) Display records
2) Add item
3) Remove item
4) Update price
5) Exit

Display records:

Input: give 1 as the input.

Expected output: displays all the food items present in the menu

Output:



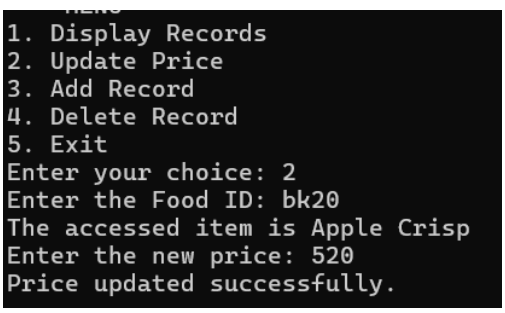
```
Food ID: bk1
Food Type: Burger
Food Name: Classic Cheeseburger
Price: 928.86
Rating: 4.31
Calories: 200
Veg: 1
No. of Times Ordered: 24
Priority: 1

Food ID: bk2
Food Type: Burger
Food Name: BBQ Bacon Burger
Price: 994.84
Rating: 4.83
Calories: 450
Veg: 0
No. of Times Ordered: 12
Priority: 1

Food ID: bk3
Food Type: Burger
Food Name: Mushroom Swiss Burger
Price: 1059.79
Rating: 4.53
Calories: 350
Veg: 1
No. of Times Ordered: 15
Priority: 1

Food ID: bk4
Food Type: Burger
Food Name: Black Bean Burger
Price: 854.88
Rating: 4.40
Calories: 400
Veg: 1
No. of Times Ordered: 8
Priority: 1
```

Update Price:

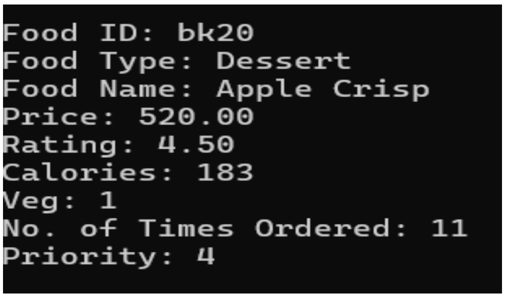
Input: Enter 2

Expected Output: price will get updated for the item

Output:



```
1. Display Records
2. Update Price
3. Add Record
4. Delete Record
5. Exit
Enter your choice: 2
Enter the Food ID: bk20
The accessed item is Apple Crisp
Enter the new price: 520
Price updated successfully.
```



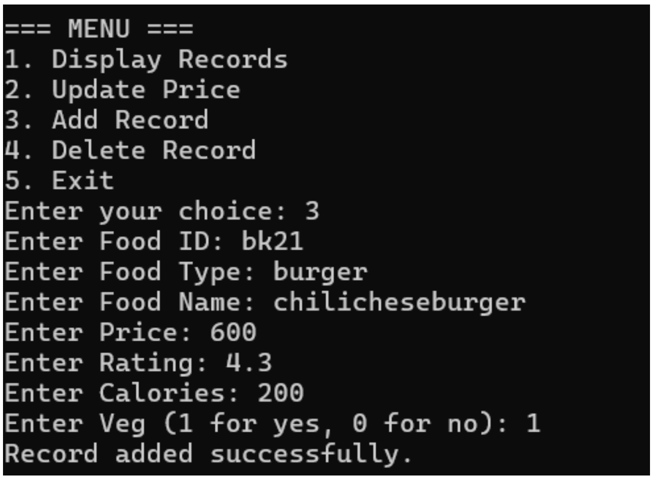
```
Food ID: bk20
Food Type: Dessert
Food Name: Apple Crisp
Price: 520.00
Rating: 4.50
Calories: 183
Veg: 1
No. of Times Ordered: 11
Priority: 4
```

Add Record:

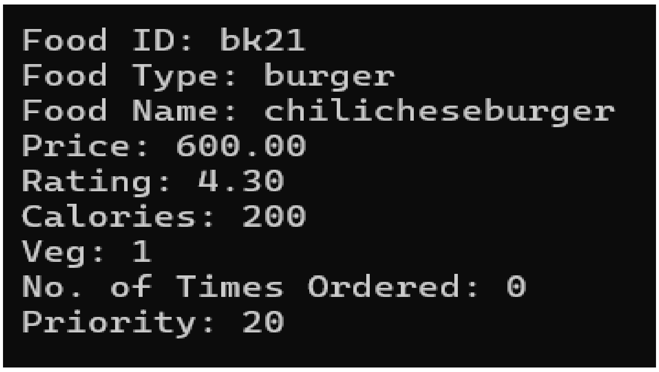
Input: enter 3 and enter the details such as food id, foodname, price, food type, rating.

Expected Output: Food item will get added to the menu.

Output:



```
=== MENU ===
1. Display Records
2. Update Price
3. Add Record
4. Delete Record
5. Exit
Enter your choice: 3
Enter Food ID: bk21
Enter Food Type: burger
Enter Food Name: chilicheseburger
Enter Price: 600
Enter Rating: 4.3
Enter Calories: 200
Enter Veg (1 for yes, 0 for no): 1
Record added successfully.
```



```
Food ID: bk21
Food Type: burger
Food Name: chilicheseburger
Price: 600.00
Rating: 4.30
Calories: 200
Veg: 1
No. of Times Ordered: 0
Priority: 20
```

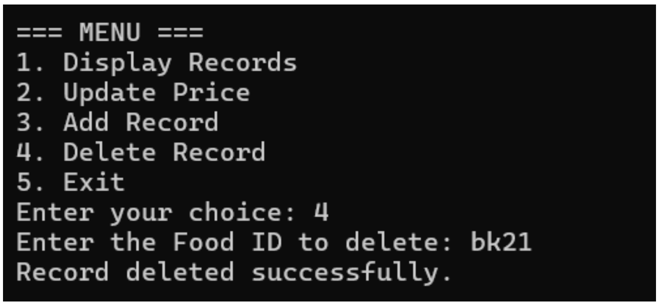
Delete Record:

To delete the item from the menu present.

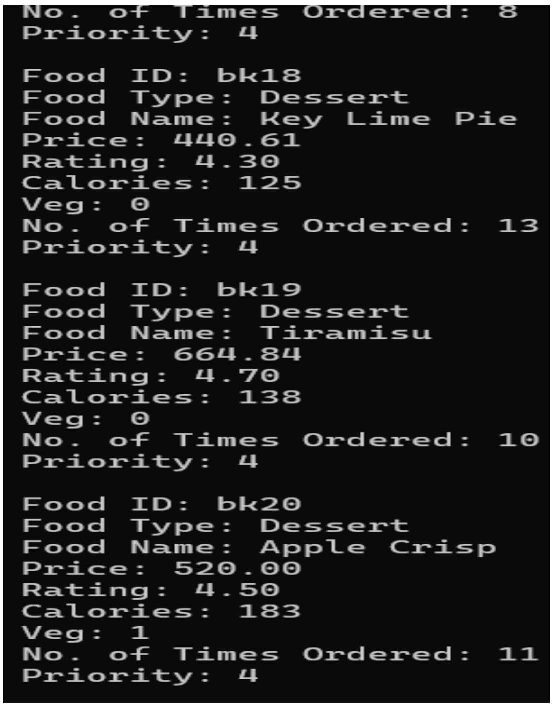
Input: Enter 4 and the food id to be removed.

Expected Output: food item will be removed from the menu.

Output:



```
=== MENU ===
1. Display Records
2. Update Price
3. Add Record
4. Delete Record
5. Exit
Enter your choice: 4
Enter the Food ID to delete: bk21
Record deleted successfully.
```



```
No. of Times Ordered: 8
Priority: 4

Food ID: bk18
Food Type: Dessert
Food Name: Key Lime Pie
Price: 440.61
Rating: 4.30
Calories: 125
Veg: 0
No. of Times Ordered: 13
Priority: 4

Food ID: bk19
Food Type: Dessert
Food Name: Tiramisu
Price: 664.84
Rating: 4.70
Calories: 138
Veg: 0
No. of Times Ordered: 10
Priority: 4

Food ID: bk20
Food Type: Dessert
Food Name: Apple Crisp
Price: 520.00
Rating: 4.50
Calories: 183
Veg: 1
No. of Times Ordered: 11
Priority: 4
```

Driver related modules:

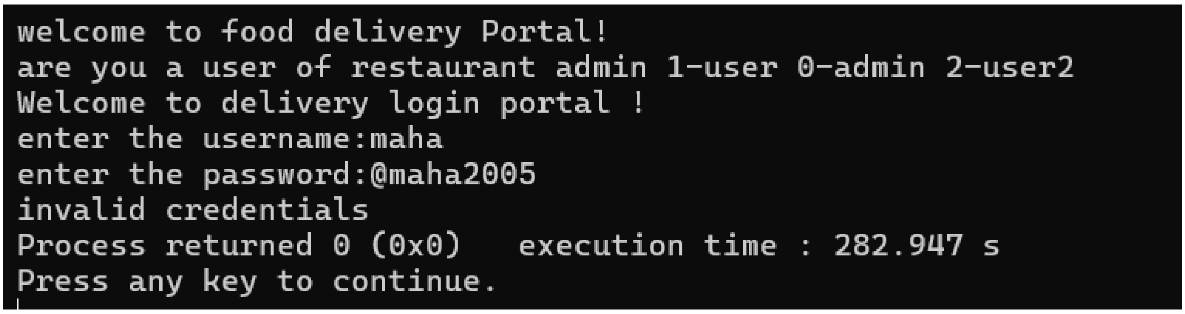
These are the following modules which the driver can operate.

Driver Login:

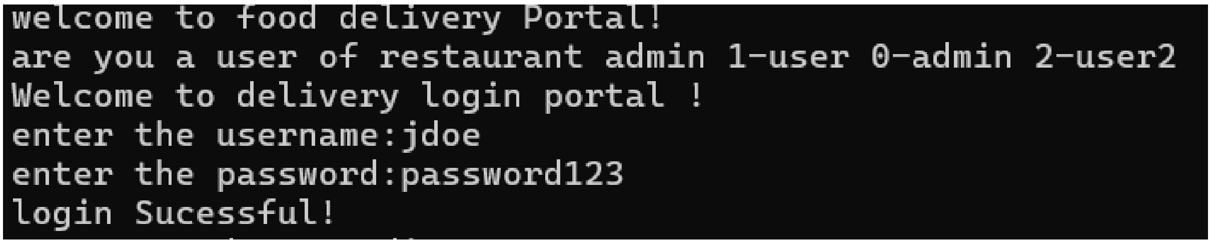
Input: Enter username and password.

Expected Output: says credentials are correct and proceeds to the next module. Else it says invalid credentials.

Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin 2-user2
Welcome to delivery login portal !
enter the username:maha
enter the password:@maha2005
invalid credentials
Process returned 0 (0x0)   execution time : 282.947 s
Press any key to continue.
```

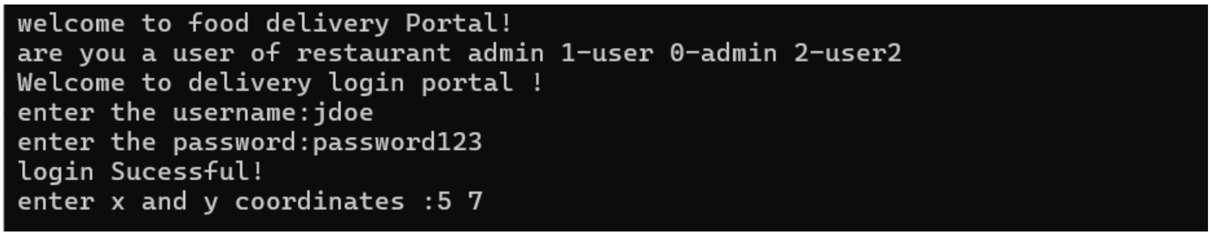


```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin 2-user2
Welcome to delivery login portal !
enter the username:jdoe
enter the password:password123
login Sucessful!
```

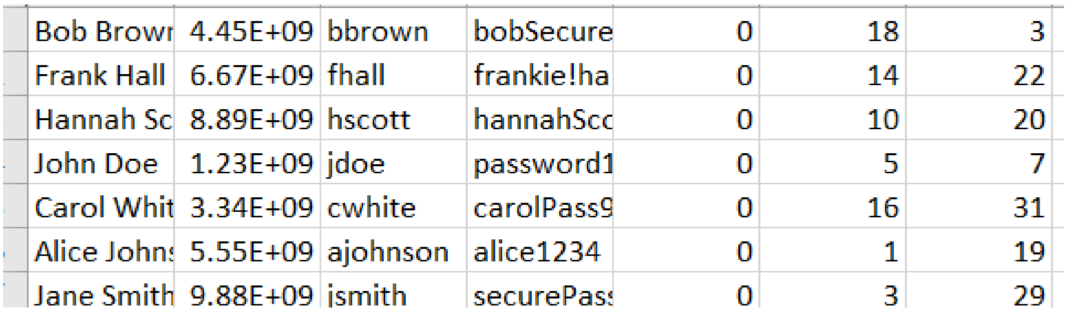
Change coordinates:

In this function the delivery guy is given the option of changing his position according to his own needs.

Input: enters the x and y coordinates.
Expected Output: changes the coordinates present in the delivery drivers database.
Output:



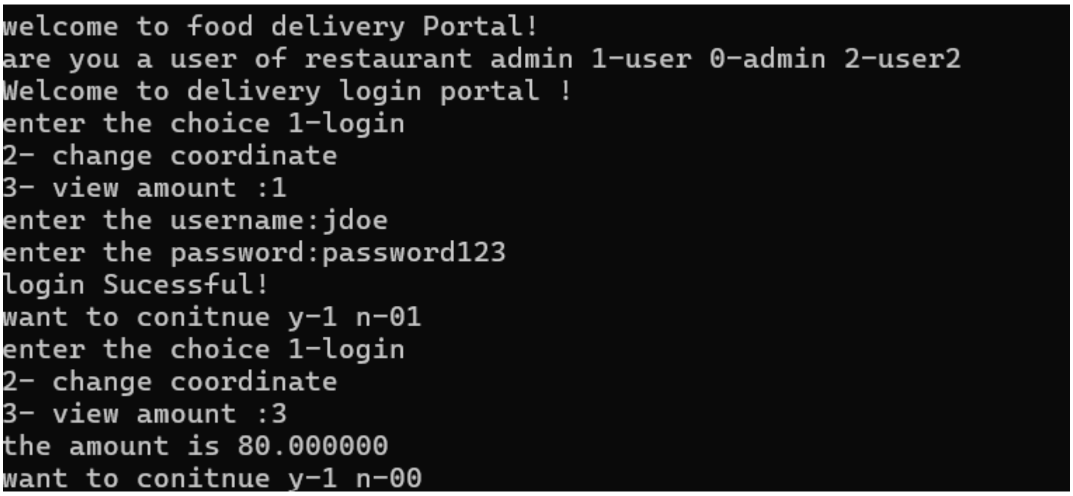
```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin 2-user2
Welcome to delivery login portal !
enter the username:jdoe
enter the password:password123
login Sucessful!
enter x and y coordinates :5 7
```



Bob Browr	4.45E+09	bbrown	bobSecure	0	18	3
Frank Hall	6.67E+09	fhall	frankie!ha	0	14	22
Hannah Sc	8.89E+09	hscott	hannahScc	0	10	20
John Doe	1.23E+09	jdoe	password1	0	5	7
Carol Whit	3.34E+09	cwhite	carolPass9	0	16	31
Alice Johns	5.55E+09	ajohnson	alice1234	0	1	19
Jane Smith	9.88E+09	jsmith	securePass	0	3	29

Display Amount:

Input: None.
Expected Output: displays the amount for the corresponding user.
Output:



```
welcome to food delivery Portal!
are you a user of restaurant admin 1-user 0-admin 2-user2
Welcome to delivery login portal !
enter the choice 1-login
2- change coordinate
3- view amount :1
enter the username:jdoe
enter the password:password123
login Sucessful!
want to conitnue y-1 n-01
enter the choice 1-login
2- change coordinate
3- view amount :3
the amount is 80.000000
want to conitnue y-1 n-00
```

LIMITATIONS OF THE SOLUTION:

- The solution uses a terminal based interface for device-user interaction.
- The live tracking of delivery personnel is not possible.
- There is no real time map used for the shortest path.
- No options for payment through popular gateways such as Gpay, Paytm.
- The data used is synthetic and limited.
- We do not take the driver status in the delivery assignment algorithm, we assume that there is only one order going on at a time.
- The solution cannot handle multiple user requests at the same time.

OBSERVATIONS (from Societal, Legal, Environmental and Ethical perspectives):

- Through the medium of the calorie tracking feature, we will inculcate healthy eating habits for the users of this application.
- Through the recommendation feature which uses the Apriori algorithm, we suggest foods which the user will not waste, thus reducing the food waste generated.
- Our partnered restaurants are encouraged to use sustainable packaging instead of plastic packaging.
- We ensure that our restaurant partners comply with relevant food standards placed by the FSSAI.
- Our drivers are employed and are given benefits such as life insurance and vehicle insurance.

LEARNING OUTCOMES:

- Through the means of this project, we learnt how to work as a team to achieve a common goal.
- We learnt how to integrate and run multiple C files, especially the use of header files.
- We learnt File handling in C.
- We learnt the use of algorithms such as the Floyd-Warshall algorithm (used to calculate shortest distance).
- We learnt the concept of associative mining (used to recommend food) and its implementation in programs.

REFERENCES:

- YouTube
- Geeks for geeks
- Swiggy
- Zomato